AD-A062 012 NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA F/G 12/2
SOLUTION OF THE INTEGER CONCAVE PROGRAM USING THE IC PHI N ALGO--ETC(U)
NOV 78 H W LOOMIS
UNCLASSIFIED NSWC/DL-TR-3120-VOL-2 NL

1 OF 2
AD
A062 012

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

14 ~~1. REPORT NUMBER~~ NSWC/DL-TR-3120-Vol-2	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9
6 ~~4. TITLE (and Subtitle)~~ Solution of the Integer Concave Program Using the ICØN Algorithm, Volume 2. IC ≠ phi N		~~5. TYPE OF REPORT & PERIOD COVERED~~ FINAL ≠ rept.
		~~6. PERFORMING ORG. REPORT NUMBER~~
7. AUTHOR(s) 10 Harlan W. Loomis		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Surface Weapons Center (K30) Dahlgren, Virginia 22448		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61152N, 16 ZR00001 17 ZR01407, NWL/01K07
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Surface Weapons Center (K30) Dahlgren, Virginia 22448		11 ~~12. REPORT DATE~~ 1 November 1978
		~~13. NUMBER OF PAGES~~ 149
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) 12 135p.		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

algorithm, branch-and-bound, computer program, concave function, convex function, integer programming, linear programming, mathematical programming, nonlinear programming, operations research, optimization theory, systems analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The branch-and-bound technique has been the basis for algorithms to solve both the mixed integer linear program (having linear objective function, linear constraints, and integrality restrictions on some variables) and the concave nonlinear program (having a separable concave objective function and linear constraints). The subprograms for each of these branch-and-bound algorithms are linear programs with simple upper bounds.

next page (Continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391 598

JB

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

In Volume 1, a new branch-and-bound algorithm is presented for the composite mixed integer, concave nonlinear program. This integer concave (ICØN) algorithm has been implemented in the form of a computer program coded in FORTRAN. A guide to the use of the computer program together with examples of its application are included in Volume 1. Documentation of the computer program is included in Volume 2.

Ø

zero

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

ABSTRACT

The branch-and-bound technique has been the basis for algorithms to solve both the mixed integer linear program (having linear objective function, linear constraints, and integrality restrictions on some variables) and the concave nonlinear program (having a separable concave objective function and linear constraints). The subprograms for each of these branch-and-bound algorithms are linear programs with simple upper bounds.

In Volume 1, a new branch-and-bound algorithm is presented for the composite mixed integer, concave nonlinear program. This integer concave (ICØN) algorithm has been implemented in the form of a computer program coded in FORTRAN. A guide to the use of the computer program together with examples of its application are included in Volume 1. Documentation of the computer program is included in Volume 2.

FOREWORD

Optimization problems involving integer-valued decision variables occur frequently in operations research and in systems analysis. Many cost-effectiveness analyses performed within the Department of Defense are optimization problems of this type. Specific applications related to amphibious operations occur in mine warfare, logistics, and fire support. This report presents a new method for solving a large class of integer nonlinear optimization problems.

The material presented here implements ideas developed during the author's program of studies in the Department of Operations Research, School of Engineering and Applied Science, The George Washington University, Washington, D. C. Support for this research in integer nonlinear optimization and the development of computational algorithms has been provided by the Naval Surface Weapons Center's Independent Research Program. The author is presently involved in surface warfare applications of this optimization technique.

RELEASED BY:

[illegible]

RALPH A. NIEMANN, Head
Warfare Analysis Department

CONTENTS

LIST OF FIGURES

LIST OF EXHIBITS

SOLUTION OF THE INTEGER CONCAVE PROGRAM USING THE ICØN ALGORITHM

VOLUME 2

1. INTRODUCTION

The ICØN branch-and-bound algorithm has been implemented in the form of a computer program coded in FORTRAN for the CDC 6700 computer system. Appendices A and B of Volume 1 describe the input data and the user subroutines which are required by the program. Appendix C of Volume 1 provides descriptive examples of input data and user subroutines for three test problems. Volume 2 includes that additional information which is needed to fully document the computer program.

Although the computer program for the ICØN algorithm was developed for use on the CDC 6700 computer, a special effort was made to assure that the programming techniques utilized would be compatible with a wide variety of computers. Minimal effort should be required to convert the program for use on other computers having a FORTRAN compiler.

The ICØN algorithm, like many branch-and-bound algorithms, requires substantial amounts of computer storage for most efficient operation. In the CDC 6700 computer system, the availability of a random access mass storage device fulfills this requirement. The CDC 6700 system subroutines which are required in order to utilize this capability are described in this report. Other computers possess similar capabilities to which the computer program can be readily adapted.

Appendix A provides the general flow schematics on which the computer program is based. Appendix B gives the definitions of parameters and arrays utilized in the program. Aspects of programming technique are discussed in Appendices C and D. Appendix C deals with the use of mass storage for the branch-and-bound list while Appendix D deals with the representation and solution of subprograms. The subroutines which comprise the ICØN algorithm as well as the system subroutines which are used by the algorithm are described in Appendix E. Finally, Appendix F provides a listing of the computer code.

APPENDIX A

FLOW SCHEMATICS

The general flow schematics for the ICØN branch-and-bound algorithm are presented in this appendix. Figure 1 shows the basic sequence of steps which are performed when Method 1 is used, and Figure 2 shows the sequence when Method 2 is used. Methods 1 and 2 are discussed in Volume 1.

The general flow schematics shown in Figures 1 and 2 agree exactly for phase 2 of the branch-and-bound algorithm. The schematic of Figure 1 can be regarded as being embedded in the schematic of Figure 2 if the test "Phase = 1 ?" in boxes 6, 18 and 28 of Figure 2 is replaced by the test "Phase = 1 and the variable best upper bound method is in use ?" With this replacement, Figure 2 represents a general flow schematic for the computer program implementing the ICØN algorithm.

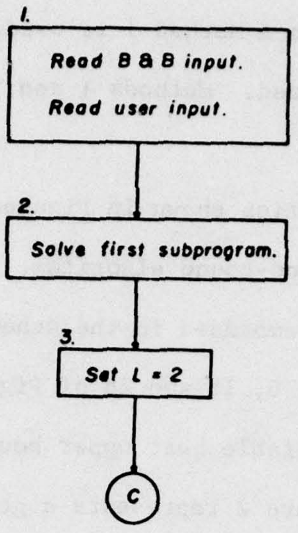


FIGURE 1

FLOW SCHEMATIC FOR THE ICØN BRANCH-AND-BOUND ALGORITHM

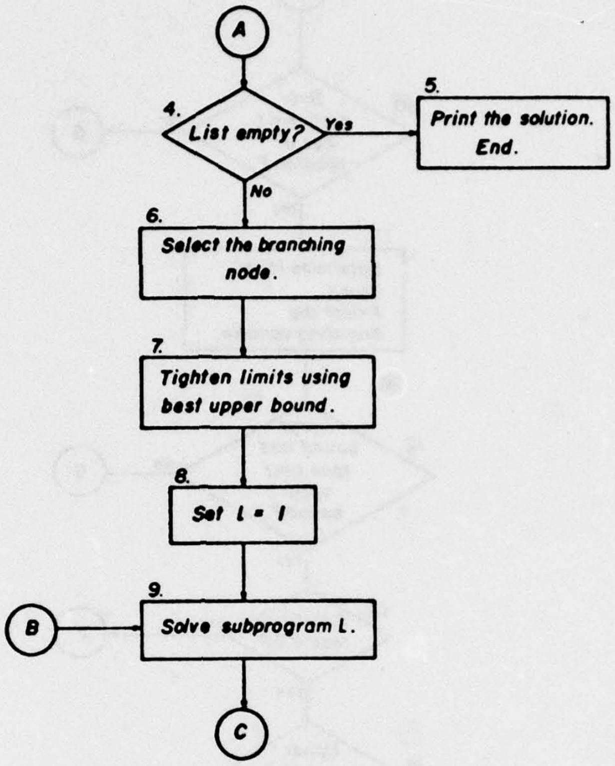


FIGURE 1 (Continued)

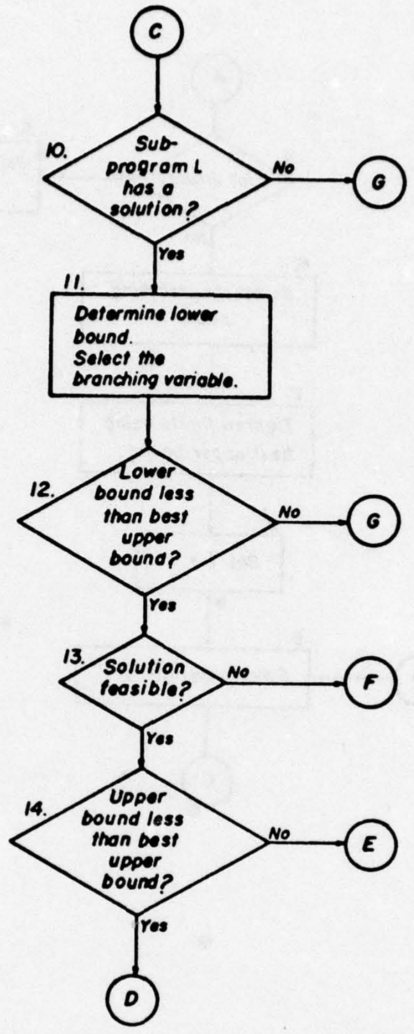

C
10.
Sub-program L has a solution?
No
G
Yes
11.
Determine lower bound. Select the branching variable.
12.
Lower bound less than best upper bound?
No
G
Yes
13.
Solution feasible?
No
F
Yes
14.
Upper bound less than best upper bound?
No
E
Yes
D

FIGURE 1 (Continued)

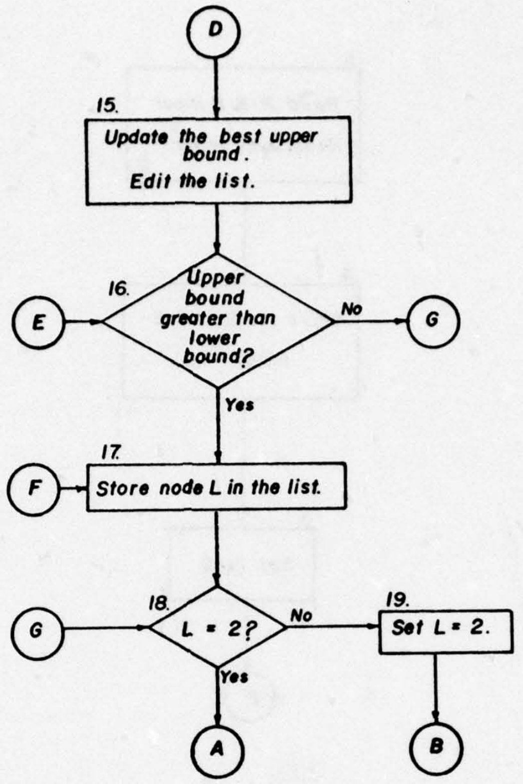

D
15.
Update the best upper bound.
Edit the list.
16.
Upper bound greater than lower bound?
No
Yes
E
G
17.
F
Store node L in the list.
18.
G
L = 2?
No
Yes
19.
Set L = 2.
A
B

FIGURE 1 (Continued)

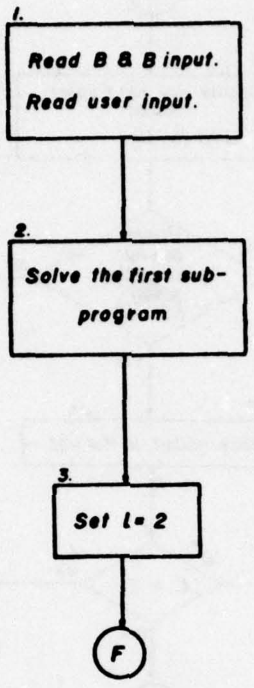
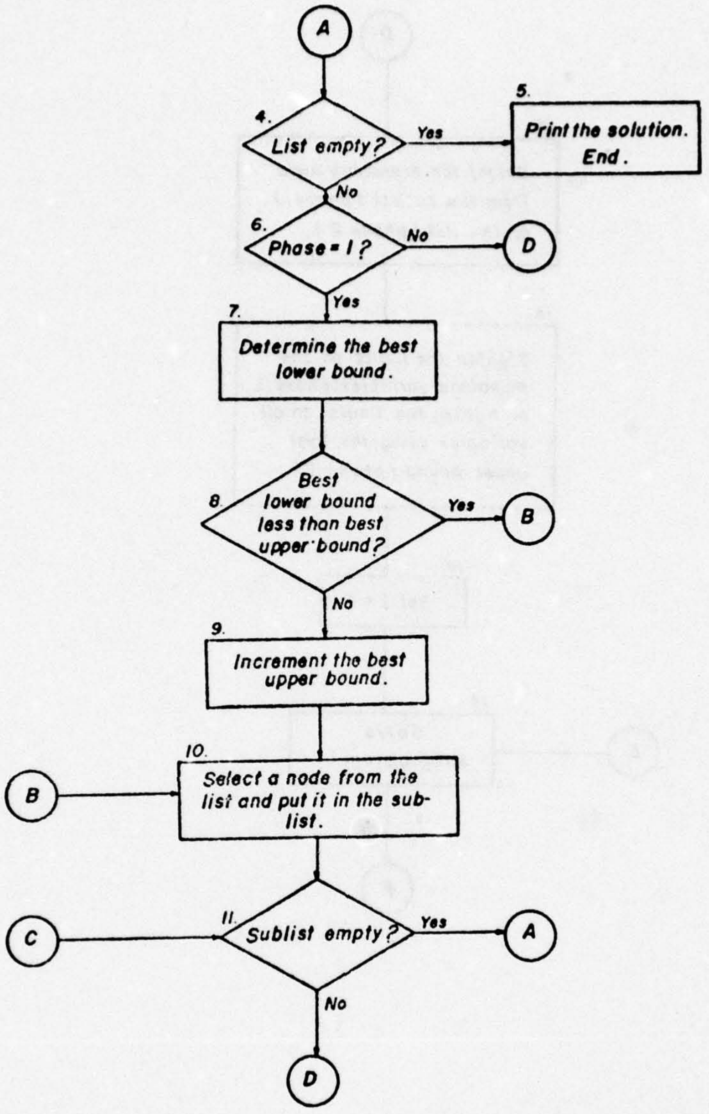


FIGURE 2

FLOW SCHEMATIC FOR THE ICØN BRANCH-AND-BOUND ALGORITHM
USING THE VARIABLE BEST UPPER BOUND METHOD



A
4.
List empty?
Yes
5.
Print the solution. End.
No
6.
Phase = 1?
No
D
Yes
7.
Determine the best lower bound.
8.
Best lower bound less than best upper bound?
Yes
B
No
9.
Increment the best upper bound.
10.
B
Select a node from the list and put it in the sub-list.
11.
C
Sublist empty?
Yes
A
No
D

FIGURE 2 (Continued)

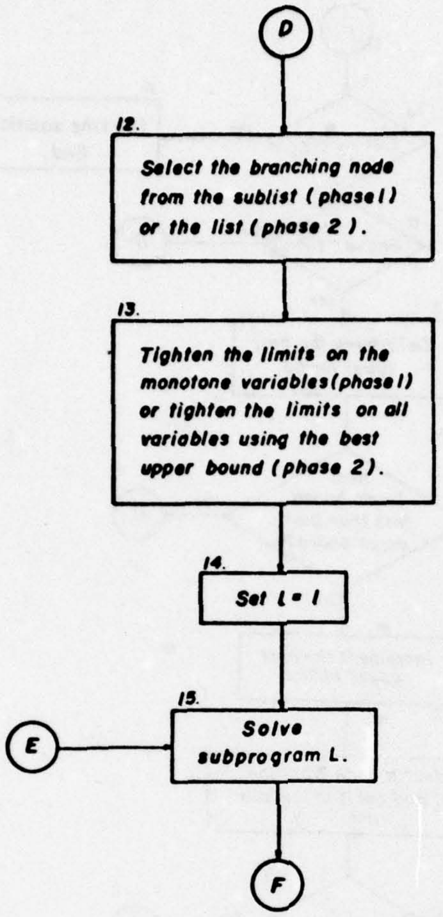

D
12.
Select the branching node from the sublist (phase I) or the list (phase 2).
13.
Tighten the limits on the monotone variables (phase I) or tighten the limits on all variables using the best upper bound (phase 2).
14.
Set L = 1
15.
Solve subprogram L.
E
F

FIGURE 2 (Continued)

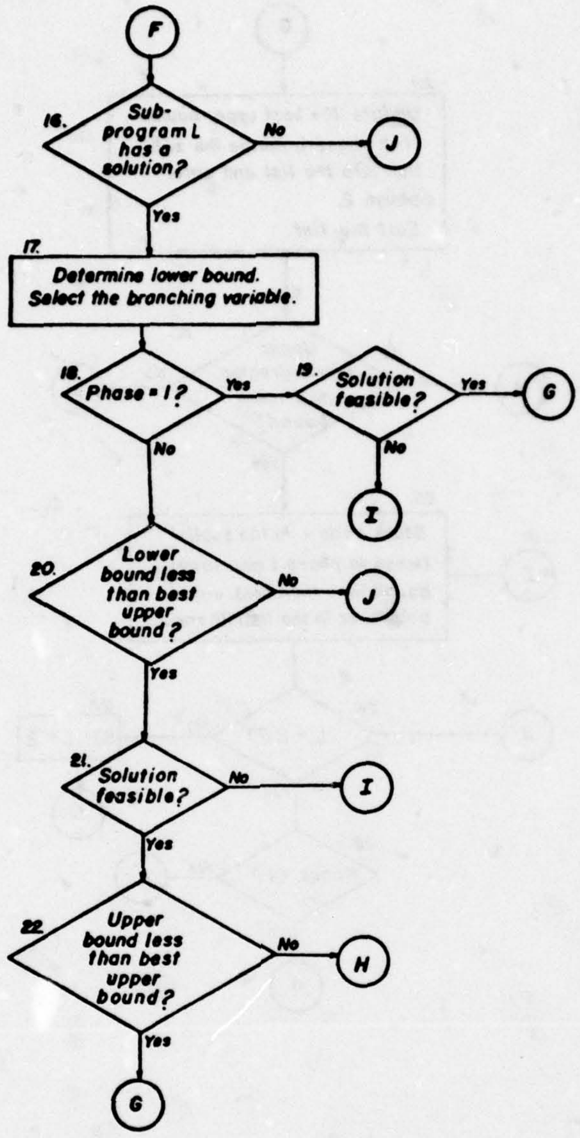
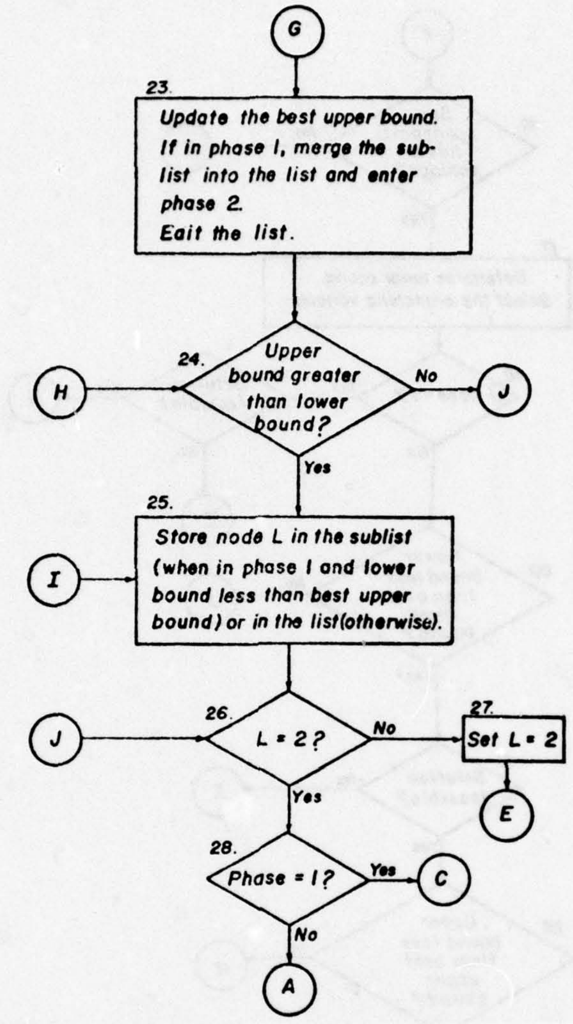

F
16.
Sub-program L has a solution?
No
J
Yes
17.
Determine lower bound. Select the branching variable.
18.
Phase = 1?
Yes
19.
Solution feasible?
Yes
G
No
I
No
20.
Lower bound less than best upper bound?
No
J
Yes
21.
Solution feasible?
No
I
Yes
22.
Upper bound less than best upper bound?
No
H
Yes
G

FIGURE 2 (Continued)



G
23.
Update the best upper bound.
If in phase I, merge the sub-list into the list and enter phase 2.
Eait the list.
24.
Upper bound greater than lower bound?
No
Yes
H
J
25.
Store node L in the sublist (when in phase I and lower bound less than best upper bound) or in the list(otherwise).
I
26.
L = 2?
No
Yes
J
27.
Set L = 2
E
28.
Phase = 1?
Yes
C
No
A

FIGURE 2 (Continued)

APPENDIX B

PARAMETERS AND ARRAYS

The parameters and arrays used in the ICØN branch-and-bound algorithm are described in this appendix. Fixed storage consists of all parameters and two arrays (arrays IMASS2 and IMASS3) which are located in labeled common storage. All other arrays are given variable dimensions, with the actual value of these dimensions being assigned during program execution.

The use of variable dimensions results in the minimization of the core storage required during a computer run, an important factor in the CDC 6700 computer system where the cost of a computer run is proportional to the amount of core storage used. A further benefit from variable dimensioning is the elimination of artificial limits on individual problem parameters (such as the number of variables, N, or the number of constraints, M) which typically result when fixed dimensions are assigned in a computer code. The use of variable dimensioning thus results both in reduced running cost and in program flexibility, where either small or large problems can be solved with the only changes being in the user supplied subroutines.

FIXED STORAGE

Exhibit 1 shows the labeled commons used in the ICØN branch-and-bound algorithm. CØMMØN/P0/ consists of a set of pointers used in conjunction with the variable dimensioning of arrays (parameters NI1 through NI12 and parameters NF1 through NF24) and in conjunction with the branch-and-bound list (parameters NIMS2 and NFMS3). CØMMØN/P1/ consists of the control parameters discussed in Appendix A, Volume 1.

CØMMØN/P2/ consists of those parameters which are used repeatedly in the algorithm. These are:

Variable	Description
EPSI and EPSIM	Tolerances (10^{-11} and -10^{11}) used to test if a number is zero within roundoff error
BIGN	A large number (10^{100}) used in place of $+\infty$
BEGTM	The clock time at which the branch-and-bound algorithm commenced the processing of a given program
M1, M2 and M3	The number of less than or equal, equality, and greater than or equal constraints in the program[1]
M4	= M2 + M3, the number of artifical variables associated with the program
N1	= N + (M3 + M1) + (M2 + M3), the total number of variables associated with the program including slack, surplus and artifical variables
MP1	= M + 1, the number of basic variables in a one phase linear program, and the position of the linear programming phase 2 objective function among the basic variables

[1]See the discussion of the constraint matrix and right-hand-side vector in Appendix A, Volume 1.

EXHIBIT 1

LABELED COMMON STORAGE

```
      COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,
     1          NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,
     2          NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,
     3          NF23,NF24,NFMS3
      COMMON/P1/N,M,ITYPE,NSTPAT,NODPL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      COMMON/P4/SAVE,KBRAN,X1
      COMMON/P5/IROUND
      COMMON/A0/IMASS2(1001),IMASS3(1001)
```

Variable	Description
MP2	= M + 2, the number of basic variables in a two phase linear program, and the position of the linear programming phase 1 objective function among the basic variables
NM3	= N + M3, a parameter used in subroutines GETCØL and ØBJ1
NM1M2	= N - M1 - M2, a parameter used in subroutine GETCØL
NM1M3	= N + M1 + M3, the total number of variables excluding the artificial variables
N1P2	= N1 + 2, the total number of variables plus the linear programming phase 1 and phase 2 objective functions
NP1	= N + 1, a parameter used in subroutines BØX2, BØX15 and ØBJ1
NSUM	The total number of nonzero entries in the constraint matrix[1]
NTC	The number of constants in the table of constants for the constraint matrix[1]
M10	The position of the objective function (either MP1 or MP2) in the current phase of the linear program

CØMMØN/P3/ consists of those parameters which are required throughout the branch-and-bound algorithm. These are:

Variable	Description
NØDNØT	The node number corresponding to the current best upper bound
UNØT	The current best upper bound

[1]See the discussion of the constraint matrix and right-hand-side vector in Appendix A, Volume 1.

Variable	Description
IBUBØP	Indicator for the variable best upper bound method (0 = do not use the method; 1 = use the method)
LPHASE	Current phase of the branch-and-bound algorithm (1 = no feasible point has yet been determined; 2 = a feasible point has been determined and the current best upper bound represents a feasible point)
NØDRUL	The current node selection rule according to the phase of the branch-and-bound algorithm
NBVRUL	The current branching variable selection rule according to the phase of the branch-and-bound algorithm
NIGHT	The current limit tightening rule according to the phase of the branch-and-bound algorithm
NLIST	Number of nodes currently in the branch-and-bound list
NLISTS	Number of nodes currently in the branch-and-bound sublist
NFEAS	The current total number of nodes for which the corresponding subprogram required complete solution
LSTMX	The current maximum size attained by the branch-and-bound list
ITRTØT	The current total number of linear programming iterations performed
ITRMAX	The current maximum number of linear programming iterations performed along any single branch of the branch-and-bound tree
BLB	The best lower bound
NBRNØD	The node number of the branching node
PBRNØD	The processing order number associated with the branching node

Variable	Description
NBRVAR	The branching variable associated with the branching node
NUPDWN	The direction for continued branching (when the node selection rule is the LIFO rule) for the branching node
XBRNØD	The value of the branching variable in the solution corresponding to the branching node
TBRNØD	The constant associated with the subprogram for the branching node
NØDE	The current node number
LNØDE	Indicator for the current node (1 = the lower node emanating from the branching node; 2 = the upper node emanating from the branching node)
Z	The optimal objective function value in a subprogram
BØUNDL	The lower bound for the current node
BØUNDU	The upper bound for the current node
TSIG	The constant associated with the subprogram for the current node
IFEAS	Indicator as to the feasibility with respect to the master problem of the solution to the current subprogram (0 = not feasible; 1 = feasible)
IBRVR1	Branching variable selection under the first branching variable selection strategy
IUPDN1	The direction for continued branching corresponding to branching variable IBRVR1
XBRVR1	The value of the branching variable IBRVR1
IBRVR2	Branching variable selection under the second branching variable selection strategy
IUPDN2	The direction for continued branching corresponding to branching variable IBRVR2

Variable	Description
XBRVR2	The value of the branching variable IBRVR2
L10	The number of nonbasic variables associated with the current subprogram tableau
NITER	A counter for the number of linear programming iterations which have been performed to reach the current subprogram tableau
NBINV	A counter, similar to NITER, for the number of linear programming interactions which have been performed since the last basis reinversion
M7	The number of basic variables associated with the current subprogram tableau (either M + 1 or M + 2)
IPHASE	Indicator of the method being applied to solve the current subprogram (1 = one phase method; 2 = two phase method)
NPHASE	The current phase of the method being applied to solve the current subprogram (when IPHASE = 1, NPHASE is not needed and is set to 0; when IPHASE = 2, NPHASE is set to 1 or 2)
NM3M7	= N + M3 + M7, the number of basic and nonbasic variables associated with the current subprogram tableau
IALGØ	The linear programming algorithm to be applied to move to an optimal tableau for the current subprogram (1 = primal algorithm; 2 = dual simplex algorithm)
IEØJ	Indicator for the tableau resulting from the application of a linear programming algorithm (0 = optimal; 1 = primal infeasible; 2 = primal unbounded; 3 = dual value exceeds the current best upper bound; 4 = maximum number of linear programming iterations exceeded)

CØMMØN/P4/ consists of three parameters which are used as temporary storage in subroutine BØX15. CØMMØN/P5/ consists of a single parameter which is set in subroutine INPUT3 and used in subroutine BØX17. It is used to indicate whether or not the objective function is an integer valued function for a mixed integer linear program. CØMMØN/AO/ consists of two arrays used in conjunction with the branch-and-bound list.

VARIABLE STORAGE

Arrays IF and F are assigned fixed dimensions in program MAIN compatible with the program or programs to be solved. These two arrays are in turn subdivided within the branch-and-bound algorithm into the various integer arrays and floating point arrays required to solve a program. This subdivision varies from program to program depending upon the program structure as specified in the input.

Exhibit 2 shows how the variable dimensioning of arrays is accomplished. Program MAIN transfers control to subroutine ICØN, at the same time passing the locations of arrays IF and F together with the corresponding dimensions NI and NF as shown at line 1 of subroutine ICØN. These two arrays are dimensioned at line 18 of subroutine ICØN. At lines 24-25, control is transferred to subroutine BØX1 which reads the input for the branch-and-bound algorithm and allocates the storage occupied by the arrays IF and F to the various integer and floating point arrays needed in the branch-and-bound algorithm. This allocation is accomplished by developing dimensions ND1 through ND11 for these arrays and by developing pointers NI1 through NI12 for the integer arrays and pointers NF1 through NF24 for the floating point arrays. The subsequent use of these variable dimensioned arrays is exemplified by the call to subroutine BØX7 at line 43 of subroutine ICØN. The first location of the eighth integer array is IF(NI8) and the first location of the thirteenth floating point array is F(NF13). It happens that both of these arrays have the same dimension, ND10. The nomenclature

EXHIBIT 2

EXAMPLE OF VARIABLE DIMENSIONING

```
      SUBROUTINE ICON (IF,F,NI,NF)                                      ICON0001
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.           ICON0002
      COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,     ICON0003
     1          NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,    ICON0004
     2          NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22, ICON0005
     3          NF23,NF24,NFMS3                                         ICON0006
      .                                                                 .
      .                                                                 .
      .                                                                 .
      DIMENSION IF(NI),F(NF)                                            ICON0018
      .                                                                 .
      .                                                                 .
  100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,   ICON0024
     1           ND11,NDMS2,NDMS3)                                      ICON0025
      .                                                                 .
      .                                                                 .
      CALL BOX7 (IF(NI8),F(NF13),ND10)                                  ICON0043
      .                                                                 .
      .                                                                 .
      END                                                               ICON0098
      SUBROUTINE BOX7 (INUSE,CAPP,ND10)                                 BOX70001
C DETERMINE THE BEST LOWER BOUND.                                       BOX70002
      .                                                                 .
      .                                                                 .
      DIMENSION INUSE(ND10)                                             BOX70013
      DIMENSION CAPP(ND10)                                              BOX70014
      .                                                                 .
      .                                                                 .
      .                                                                 .
      END                                                               BOX70024
```

associated with these arrays is modified as shown at line 1 of subroutine BØX7. The labels INUSE and CAPP are mnemonics which reflect the functions performed by these arrays within the branch-and-bound algorithm. The arrays are dimensioned at lines 13-14 of subroutine BØX7 and are manipulated within this subroutine in a manner no different from that of an array having fixed dimensions.

The dimensions ND1 through ND11 are established in subroutine BØX1 from the control parameters discussed in Appendix A, Volume 1. The values of these dimensions are as follows:

ND1	= N	
ND2	= NSUM	
ND3	= NTC	
ND4	= M + 2	
ND5	= N + M3	
ND6	= M + 2 + N + M3	
ND7	= M + 2	(= 1 if the basis reinversion feature of the program is not used, that is if IFB = MBINV = LISTØP = 0)
ND8	= M + 2	(= 1 if the sensitivity slopes are not needed, that is if NSTRAT = 1, NBVRL1 $\geq$ 3 and NTITE1 = 1, or if NSTRAT = 2, NBVRL1 $\geq$ 3, NBVRL2 $\geq$ 3 and NTITE1 = NTITE2 = 1)
ND9	= N	(=1 if the program to be solved is a mixed integer linear program or a linear program, that is if ITYPE = 1 or 3)
ND10	= MXLIST	

ND11 = MXLIST	(= 1 if the LIFO node selection rule is not used, that is if NSTRAT = 1 and NØDRL1 = 1 or if NSTRAT = 2 and NØDRL1 = NØDRL2 = 0).

Array IF is subdivided into the following twelve integer arrays:

Array (Dimension)	Description
1. NZ(ND1)	The number of nonzero entries in the constraint matrix by column
2. NP(ND1)	Pointers marking the beginning of each column for arrays IR and IA
3. IR(ND2)	Row index for a nonzero entry in the constraint matrix
4. IA(ND2)	Pointer to the appropriate constant (in the table of constants) for a nonzero entry in the constraint matrix
5. INT(ND1)	Markers for integer variables (0 = not integer; 1 = first integer variable; 2 = second integer variable; etc.)
6. ICC(ND1)	Markers for concave variables (0 = not concave; 1 = first concave variable; 2 = second concave variable; etc.)
7. IS(ND4)	Temporary storage used in the transfer of one column of the constraint matrix (corresponds to the data in array IR)
8. INUSE(ND10)	Indicator for an entry in the branch-and-bound list (0 = not in use; positive integer = node number of active node in the list; negative integer = node number of active node in the sublist)
9. NV(ND6)	Temporary storage
10. IBV(ND4)	Current list of basic variables in a subprogram
11. NBV(ND5)	Current list of nonbasic variables in a subprogram

Array (Dimension)	Description
12. IUPPER(ND5)	Upper bound indicator for nonbasic variables given in array NBV (0 = nonbasic variable is at lower bound; 1 = nonbasic variable is at upper bound)

Array F is subdivided into the following 24 floating point arrays:

Array (Dimension)	Description
1. TC(ND3)	The table of constants for the constraint matrix
2. BØRIG(ND4)	The original right-hand-side vector specified in the program input
3. RHS(ND4)	The initial right-hand-side vector for a subprogram
4. C2(ND1)	The coefficients of the program objective function for variables which enter linearly (i.e., objective function values for concave variables are provided through subroutine GETØBJ)
5. C1(ND5)	Objective function coefficients used in phase 1 of the linear programming solution of the first subprogram
6. BI(ND4)	The current values of basic variables in a subprogram
7. BN(ND5)	The current values of nonbasic variables in a subprogram
8. U(ND6)	The upper limits on the variables in a subprogram
9. PJ(ND4)	Temporary storage used (together with array IS) in the transfer of one column of the constraint matrix
10. BINV(ND7,ND7)	Temporary storage used to develop the basis inverse when the basis reinversion feature of the program is exercised
11. XJ(ND4)	The updated column of the entering variable in the subprogram

Array (Dimension)	Description
12. XNØT(ND1)	The current best solution in the branch-and-bound algorithm
13. CAPP(ND10)	The lower bounds associated with nodes saved in the branch-and-bound list
14. CAPL(ND11)	The processing order associated with nodes saved in the branch-and-bound list for use in the LIFO node selection rule
15. SIGMAL(ND6)	The lower limits on the variables for a subprogram
16. SIGMAU(ND6)	The upper limits on the variables for a subprogram
17. V(ND6)	Temporary storage (used in conjunction with array NV)
18. XZ(ND6)	The solution to a subprogram adjusted for lower bound constraints
19. SO(ND8)	The "left" sensitivity slopes associated with the basic variables in a subprogram solution
20. S1(ND8)	The "right" sensitivity slopes associated with the basic variables in a subprogram solution
21. SLØLD(ND6)	The lower limits on the variables for the branching node
22. SUØLD(ND6)	The upper limits on the variables for the branching node
23. C2ØLD(ND9)	The coefficients of the program objective function for the branching node
24. B(ND4,ND4)	The current basis inverse in a subprogram

The pointers NI1 through NI12 are determined in subroutine BØX1 so that the elements IF(NI1) through IF(NI12) of array IF correspond to the first elements of arrays NZ through IUPPER. This is done, in a natural fashion, by setting NI1 = 1, NI2 = NI1 + ND1 where ND1 is the dimension of array NZ, NI3 = NI2 + ND1 where ND1 is the dimension of array NP, and so forth. The pointers NF1 through NF24 are similarly determined so that the elements F(NF1) through F(NF24) of array F correspond to the first elements of arrays TC through B.

APPENDIX C

BRANCH-AND-BOUND LIST

The branch-and-bound list is discussed in this appendix. The list contains a record of the active nodes in the branch-and-bound tree together with the data required to characterize each such node. Associated with the list is the data required by the branching node selection rule or rules to manipulate the list. The record of the active nodes and the branching node selection data are maintained in core storage, while the data required to characterize each node are maintained in random access mass storage.

The items of information which are maintained in core storage are the following:

(i) The number of nodes in the list (parameter NLIST);

(ii) The number of nodes in the sublist (parameter NLISTS);

(iii) The node number for each active node (array INUSE);

(iv) The lower bound for each active node, used by the priority node selection rule (array CAPP); and

(v) The processing order for each active node, used by the LIFO node selection rule (array CAPL).

The items of information which are saved in random access mass storage for each active node are the following:

(i) The lower and upper limits on the variables which serve to characterize a node (arrays SIGMAL and SIGMAU);

(ii) Branching variable selection data (parameters IBRVR1, IUPDN1, XBRVR1, IBRVR2, IUPDN2, XBRVR2);

(iii) Limit tightening information (parameter Z, arrays XZ, S0, S1); and

(iv) The optimal tableau associated with the subprogram (parameters TSIG, L10, NITER, NBINV, M7, IPHASE, NPHASE, NM3M7, arrays IBV, NBV, IUPPER, C2, B).

Depending upon the program solution strategy, the program type, and the branch-and-bound list option, certain of these items need not be saved and are omitted from the list. If only one solution strategy is to be used in the solution of the program (NSTRAT = 1), the data associated with the second solution strategy (parameters IBRVR2, IUPDN2, XBRVR2) are omitted from the list. If limit tightening is not included in the program solution strategy (NSTRAT = 1 and NTITE1 = 1, or NSTRAT = 2 and NTITE1 = NTITE2 = 1), then the data associated only with limit tightening (arrays XZ, SO, S1) are omitted from the list. If the program is a mixed integer linear program (ITYPE = 1), the objective function coefficients (array C2) are the same for all nodes and are omitted from the list. If the branch-and-bound list option indicates that the basis inverse should not be included in the list (LISTØP = 1), then the basis inverse matrix associated with the optimal tableau for a node (array B) is not saved in the list. The basis reinversion feature is then used to regenerate the basis inverse matrix in the event that this node is selected for branching.

The balance of this appendix discusses the mechanics associated with manipulating that portion of the branch-and-bound list which is maintained in random access mass storage. Four integer arrays are associated with the data to be saved in the list. These are:

NV(ND6)

IBV(ND4)

NBV(ND5)

IUPPER(ND5).

The eleven integer parameters

IBRVR1

IUPDN1

IBRVR2

IUPDN2

L10

NITER

NBINV

M7

IPHASE

NPHASE

NM3M7

are transferred into the last eleven locations of array NV. The integer parameters and arrays to be saved in the list thus occupy

$$NDMS2 = ND4 + 2 \cdot ND5 + 11$$

consecutive locations in core storage, all within the basic integer array IF. An auxiliary array IMS, having variable dimension NDMS2, is used to refer to this integer data. The first location of array IMS within the array IF is positioned eleven locations before the first location of array IBV. Setting

$$NIMS2 = NI10 - 11,$$

IMS(1) corresponds to IF(NIMS2).

Eight floating point arrays are associated with the data to be saved in the list. These are:

V(ND6)

XZ(ND6)

SO(ND8)

S1(ND8)

SLØLD(ND6)

SUØLD(ND6)

C2ØLD(ND9)

B(ND4,ND4).

The contents of arrays SIGMAL, SIGMAU and C2 are transferred into the arrays SLØLD, SUØLD and C2ØLD (respectively). The four floating point parameters

Z

TSIG

XBRVR1

XBRVR2

are transferred into the last four locations of array V. The floating point parameters and arrays to be saved in the list thus occupy

$$NDMS3 = (ND4)^2 + 3 \cdot ND6 + 2 \cdot ND8 + ND9 + 4$$

consecutive locations in core storage, all within the basic floating point array F. An auxiliary array FMS, is used to refer to this floating point data. The first location of array FMS within the array F is positioned four locations before the first location of array XZ. Setting

$$NFMS3 = NF18 - 3,$$

FMS(1) corresponds to F(NFMS3).

The definitions of the dimensions NDMS2, NDMS3 and the pointers NIMS2, NFMS3 just given correspond to the maximum possible list size. In the event that selected items of data are omitted from the branch-and-bound list, these dimensions and pointers are modified to reflect any such omissions.

Tape units 2 and 3 are the areas of mass storage occupied by the branch-and-bound list. Three CDC 6700 system subroutines are used to establish a random access structure for these areas of mass storage (subroutine ØPENMS), to transfer the data for the branching node from mass storage into core storage (subroutine READMS), and to transfer the data for an active node from core storage into mass storage (subroutine WRITMS). Exhibit 3 illustrates the use of these subroutines in the manipulation of the branch-and-bound list. At line 22 of subroutine ICØN, a call to subroutine ØPENMS establishes the random access structure for tape unit 2. Array IMASS2 is used to store subindices or pointers for 1000 "compartments" located within this area of mass storage. The call to subroutine ØPENMS at line 23 of subroutine ICØN establishes a similar structure for tape unit 3 using array IMASS3 to store subindices. Subroutine BØX1, called at lines 24-25 of subroutine ICØN, sets the values for dimensions NDMS2, NDMS3 and pointers NIMS2, NFMS3 for the particular program being solved. Subroutine BØX12, called at lines 49-51 of subroutine ICØN, selects the branching node from the branch-and-bound list (or sublist). The compartment in which

EXHIBIT 3

MANIPULATION OF THE BRANCH-AND-BOUND LIST

```
      SUBROUTINE ICON (IF,F,NI,NF)                                      ICON0001
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.            ICON0002
      COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,     ICON0003
     1          NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,    ICON0004
     2          NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22, ICON0005
     3          NF23,NF24,NFMS3                                         ICON0006
      .                                                                 .
      .                                                                 .
      COMMON/A0/IMASS2(1001),IMASS3(1001)                               ICON0017
      DIMENSION IF(NI),F(NF)                                            ICON0018
      .                                                                 .
      .                                                                 .
      CALL OPENMS (2,IMASS2,1001,0)                                     ICON0022
      CALL OPENMS (3,IMASS3,1001,0)                                     ICON0023
  100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,   ICON0024
     1           ND11,NDMS2,NDMS3)                                      ICON0025
      .                                                                 .
      .                                                                 .
  160 CALL BOX12 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),     ICON0049
     1            F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,     ICON0050
     2            ND9,ND10,ND11,NDMS2,NDMS3)                            ICON0051
      .                                                                 .
      .                                                                 .
  240 CALL BOX25 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),     ICON0086
     1            F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,     ICON0087
     2            ND9,ND10,ND11,NDMS2,NDMS3)                            ICON0088
      .                                                                 .
      .                                                                 .
      END                                                               ICON0098

      SUBROUTINE BOX12 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,     BOX10001
     1                  SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NDMS2,    BOX10002
     2                  NDMS3)                                          BOX10003
C SELECT THE BRANCHING NODE FROM THE SUBLIST (PHASE 1) OR THE LIST       BOX10004
C (PHASE 2).                                                            BOX10005
      .                                                                 .
      .                                                                 .
      DIMENSION INUSE(ND10),IMS(NDMS2)                                  BOX10016
      DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6),  BOX10017
     1          SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)             BOX10018
      .                                                                 .
      .                                                                 .
      CALL READMS (2,IMS,NDMS2,I0)                                      BOX10065
      CALL READMS (3,FMS,NDMS3,I0)                                      BOX10066
      .                                                                 .
      .                                                                 .
      END                                                               BOX10104
```

EXHIBIT 3 (Continued)

```
      SUBROUTINE BOX25 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,      BOX20001
     1                  SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NDMS2,     BOX20002
     2                  NDMS3)                                           BOX20003
C STORE NODE LNODE IN THE SUBLIST (WHEN IN PHASE 1 AND LOWER BOUND LESS BOX20004
C THAN BEST UPPER BOUND) OR IN THE LIST (OTHERWISE).                    BOX20005
      .                                                                       .
      .                                                                       .
      DIMENSION INUSE(ND10),IMS(NDMS2)                                   BOX20016
      DIMENSION C2(ND11),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6),  BOX20017
     1          SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)              BOX20018
      .                                                                       .
      .                                                                       .
      .                                                                       .
  360 CALL WRITMS (2,IMS,NDMS2,I0)                                       BOX20145
      CALL WRITMS (3,FMS,NDMS3,I0)                                       BOX20146
      .                                                                       .
      .                                                                       .
      END                                                                BOX20195
```

the data for the branching node is stored (parameter IO) is determined. At line 65 of subroutine BØX12, subroutine READMS is called to transfer the data contained in compartment IO on tape unit 2 into the array IMS in core storage. At line 66 of subroutine BØX12, a similar call of subroutine READMS transfers the data contained in compartment IO on tape unit 3 into the array FMS. Subroutine BØX25, called at lines 86-88 of subroutine ICØN, stores the data for an active node in the branch-and-bound list (or sublist). An available compartment is found (parameter IO) and, at lines 145-146 of subroutine BØX25, two calls to subroutine WRITMS transfer the data contained in arrays IMS and FMS in core storage into compartment IO on tape unit 2 and compartment IO on tape unit 3 (respectively).

APPENDIX D

SUBPROGRAM REPRESENTATION AND SOLUTION

The representation and solution of the subprograms generated by the branch-and-bound algorithm are discussed in this appendix. The data associated with a subprogram (a linear program) are:

(i) A, the M x N matrix of constraint coefficients;

(ii) RHS, the right-hand-side vector;

(iii) C1, the vector of coefficients for the (linear programming) phase 1 objective function;

(iv) C2, the vector of coefficients for the (linear programming) phase 2 objective function; and

(v) U, the vector of simple upper bounds on the program variables.

Figure 3 displays this data in the form in which it is used in the computer program.

Specific labeling conventions are adopted for the rows and columns of the linear program. The (M + 2) rows are ordered as follows:

Less than or equal constraints	(M1 rows)
Equality constraints	(M2 rows)
Greater than or equal constraints	(M3 rows)
Phase 2 objective function	(1 row)
Phase 1 objective function	(1 row).

The (N + M + M3 + 2) columns of the linear program are ordered as follows:

Program variables	(N columns)
Surplus variables	(M3 columns)
Slack variables	(M1 columns)
Artificial variables (equality constraints)	(M2 columns)

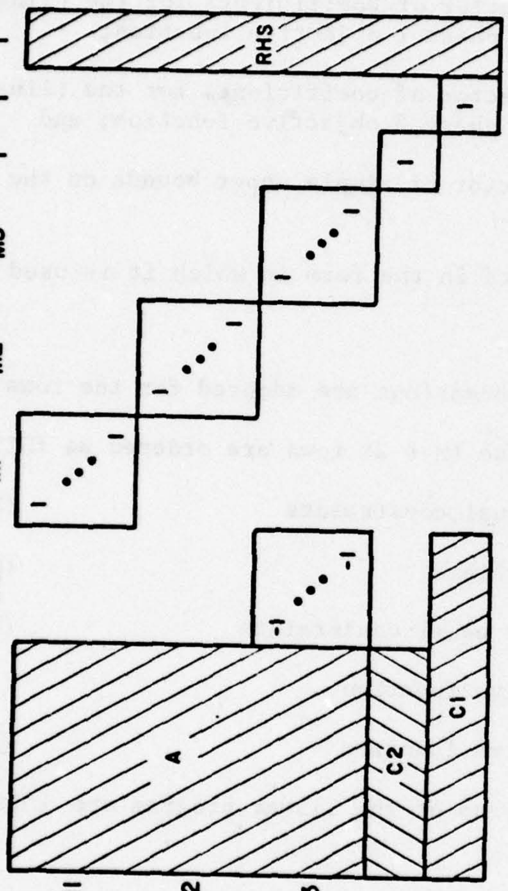


FIGURE 3

SUBPROGRAM REPRESENTATION

Artificial variables (greater than or equal constraints)	(M3 columns)
Phase 2 objective function	(1 column)
Phase 1 objective function	(1 column)
Right-hand-side	(1 column).

The constraint matrix A is represented within the computer program in a form which is commonly used for large linear programs. The density of nonzero entries in the constraint matrix is frequently low for such programs. Moreover, among the nonzero entries, particular values (such as 1 or - 1) may recur often. Maintaining the entire M x N dimensional constraint matrix in core storage is wasteful for matrices of this type. An alternative means of representing the constraint matrix begins by establishing a table of constants which lists the distinct values which occur in the matrix. For each column of the matrix, the following information is then recorded:

(i) The number of nonzero entries;

(ii) For each nonzero entry, the row number in which the entry occurs; and

(iii) For each nonzero entry, a pointer which indicates the appropriate value in the table of constants.

In the computer program, the array TC corresponds to the table of constants, the array NZ corresponds to the number of nonzero entries by column, the array IR contains the row numbers in which nonzero entries occur, and the array IA contains the corresponding pointers to the table of constants. An auxiliary array NP is defined by the relations NP(1) = 0 and

$$NP(J) = \sum_{I=1}^{J-1} NZ(I)$$

for J = 2,N. This array provides a set of pointers for use in conjunction with arrays IR and IA. The data corresponding to the J-th column of the constraint matrix is then stored in the locations NP(J) + 1 through NP(J) + NZ(J) of arrays IR and IA. For extremely large programs, the integer arrays NZ, NP, IR and IA could be maintained in core storage by packing them.

The linear programming code used in this branch-and-bound algorithm utilizes the revised simplex method with simple upper bounds on the variables. This version of the simplex method is discussed in Lasdon (1970) and in Garfinkel and Nemhauser (1972). Both the primal and the dual simplex algorithms are included. The primal algorithm operates either as a one phase method or as a two phase method depending upon the requirements of the program being solved. Associated with this linear programming code is a basis inversion feature which can be used to combat the accumulation of round-off error associated with the simplex pivot operations.

The data associated with the current tableau in a subprogram are:

(i) IBV, the vector of indices of basic variables;

(ii) NBV, the vector of indices of nonbasic variables;

(iii) IUPPER, the vector of indicators of nonbasic variables at upper bound;

(iv) BI, the vector of values of basic variables;

(v) BN, the vector of values of nonbasic variables; and

(vi) B, the basis inverse.

These arrays are updated at each simplex iteration. Note that

$$BI = B(RHS - \sum_j U_j \cdot P_j)$$

where the summation is taken over all nonbasic variables j which are at upper bound and where P_j denotes the corresponding column in Figure 3. Also, BN can be obtained directly from IUPPER and U. As a consequence, BI and BN need not be saved in the branch-and-bound list.

The initial feasible basis for the first subprogram consists of slack and artificial variables unless the user elects to provide a basis as a part of the branch-and-bound input. A phase 1 objective function is established if there are artificials in the initial basis. The basis is primal feasible if the initial values of the basic variables are nonnegative and do not exceed their upper bounds. In this case, the primal algorithm is applied to solve subprogram 1, using either a one phase method (if there are no artificials in the basis) or a two phase method (otherwise). If the initial basis is not primal feasible, the reduced costs are examined to see if they are nonnegative, in which case the initial basis is dual feasible. In this case, the dual simplex algorithm is applied to solve subprogram 1. In the event that there are artificials in the initial basis, the phase 1 objective function is treated as a constraint so that the values of these artificials are forced to be zero in the solution. When the solution to subprogram 1 is reached, the optimal basis is tested to see if it contains artificial variables. If no artificials are in this basis, the row and column corresponding to the phase 1 objective function are deleted from the tableau. If there are artificials remaining in the basis, these necessarily have value zero. However, the phase 1 objective function must be maintained in the tableau as a constraint to assure that these artificials cannot assume positive values in the solutions to subsequent subprograms.

Linear programming sensitivity analysis is applied to solve the subprograms after the first subprogram. This allows the solution of a subprogram to proceed from the optimal tableau associated with the branching node. If the basis inverse is not included in the branch-and-bound list, then it is regenerated using the basis inversion feature of the linear programming code. The simple upper bounds U are computed to be the difference between the lower and upper limits SIGMAL and SIGMAU (respectively) on the variables. The initial values BI of the basic variables are computed by adjusting RHS for variables at upper bound and then multiplying by the basis inverse. As a result of the new constraint added for the branching variable and of the tightening of lower and upper limits for the basic variables, this tableau may be primal infeasible. Since the initial feasible point was the optimal solution for the branching node, this tableau is dual feasible. The dual simplex algorithm is applied to generate an optimal solution under the new constraints. The cost coefficients C2 are next updated to reflect any changes made to the lower and upper limits for concave variables. When the cost coefficients change, the reduced costs appearing in the tableau also change resulting in a tableau which is not in canonical form. The basis inverse and values of the basic variables are modified to re-establish the canonical form of the tableau, which then is primal feasible but not optimal. The primal algorithm is applied to generate an optimal solution for the new cost data. An adequate presentation of the linear programming sensitivity analysis techniques used here may be found in Hillier and Lieberman (1967).

APPENDIX E

SUBROUTINES

The subroutines used in the ICØN branch-and-bound algorithm are discussed in this appendix. The algorithm consists of 29 subroutines. Three of these (MAIN, READIN and GETØBJ are provided by the user for each program to be solved. These are discussed in Appendices B and C of Volume 1. Following is a description of each of the remaining 26 subroutines, indicating the function performed by the subroutine in the algorithm and any notable features of the subroutine. The nomenclature selected for subroutines BØX1 through BØX25 corresponds to the labels which appear in Figure 2, the general flow schematic for the algorithm. Figure 4 shows the number of times one of these subroutines calls or is called by another subroutine.

Subroutine	Description
ICØN	This is the master subroutine which calls the various subroutines in the sequence specified in the general flow schematic for the branch-and-bound algorithm. The initiation of a program solution from previously prepared restart tapes and the preparation of restart tapes when the program time limit is reached are accomplished by calling subroutine RSTART.
BØX1	The branch-and-bound input and the user input are accepted by this subroutine by means of calls to subroutines INPUT1, INPUT2, INPUT3 and READIN. The dimensioning of arrays and the structuring of the branch-and-bound list are done in BØX1.
BØX2	This subroutine initializes the data used in the branch-and-bound algorithm. The initial basis, basis inverse and values of the basic variables are established by a call to subroutine INPUT4. The algorithm to be used to solve the first subprogram is determined by subroutine INPUT5. The first subprogram is then solved.

CALLING SUBROUTINE

SUBROUTINE CALLED	MAIN	ICØN	BØX1	BØX2	BØX5	BØX13	BØX15	BØX17	BINVRT	INPUT4	INPUT5	SIMPLE	SLØPES
ICØN	1												
BØX1		1											
BØX2		1											
BØX5		2											
BØX7		1											
BØX10		1											
BØX12		1											
BØX13		1											
BØX15		1											
BØX17		1											
BØX23		1											
BØX25		1											
ADJUST						1							
BINVRT							1			1		1	
GETCØL				1			3		3	1	1	3	1
INPUT1			1										
INPUT2			1										
INPUT3			1										
INPUT4				1									
INPUT5				1									
INVERT									1				
ØBJ1										2			
RSTART		2											
SIMPLE				1			2						
SLØPES				1			1						
TIMEC		1	2		1	1							

SUBROUTINE CALLED	MAIN	ICØN	BØX1	BØX2	BØX5	BØX13	BØX15	BØX17	BINVRT	INPUT4	INPUT5	SIMPLE	SLØPES
READIN			1										
GETØBJ				2		5	2	7					

FIGURE 4

CALLS TO ICØN SUBROUTINES

Subroutine	Description
BØX5	The solution to the problem at the end of the branch-and-bound computation is printed by this subroutine. It also prints the current best solution in the event that the program time limit is reached and restart tapes are prepared.
BØX7	This subroutine computes the current best lower bound as required by the variable best upper bound method.
BØX10	This subroutine selects a node from the branch-and-bound list and places it in the sublist as required by the variable best upper bound method.
BØX12	The branching node is selected from the sublist (in phase 1 of the variable best upper bound method) or from the list (in phase 1 of any other solution method and in phase 2). The data for the branching node is read in from random access mass storage (using subroutine READMS) if this data is not already in core. The branching variable selection is adjusted according to the current phase of the branch-and-bound algorithm.
BØX13	This subroutine tightens the limits on monotone variables during phase 1 and tightens the limits on all basic variables using the best upper bound during phase 2. The latter function is accomplished by calling the auxiliary subroutine ADJUST.
BØX15	The subprograms associated with the two nodes obtained from the branching node are solved in this subroutine. The tableau associated with the branching node is modified to reflect any new lower and upper limits on the variables. The modified tableau is the starting point for the solution of both subprograms and is saved on tape unit 4 for use in the second subprogram solution. The first subprogram is solved using the dual simplex and primal algorithms, as required. The information on tape unit 4 is read in before the second subprogram is solved.
BØX17	This subroutine determines the lower bound for the node and selects the branching variable. If one of the most noninteger branching variable selection rules (rules 3 or 4) is being used for a mixed

Subroutine	Description
	integer linear program (ITYPE = 1), or if the conventional branching variable selection rule (rule 5) is being used for a concave program (ITYPE = 2), the lower bound is taken to be the objective function value Z. If the maxmin or maxmax branching variable selection rules (rules 0, 1 or 2) are being used, the stronger lower bound corresponding to the maxmin penalty is used. If it was determined in subroutine INPUT3 that the objective function is integer valued, then this lower bound is rounded up. When a two strategy method is being used, two branch-variable selections are made.
BØX23	The best upper bound is updated in this subroutine. The branch-and-bound list is edited to delete any nodes for which the lower bound exceeds this new best upper bound. If the algorithm was in phase 1, then phase 2 is entered and the sublist is merged into the list when the variable best upper bound method is being used.
BØX25	The current node is saved in the sublist (in phase 1 of the variable best upper bound method when the lower bound is less than the best upper bound) or in the list (otherwise). The data for the node is placed in random access mass storage (using subroutine WRITMS). In the event that the maximum list size is exceeded, the node having the greatest lower bound is erased from the list to make room for the new node.
ADJUST	This subroutine is called by subroutine BØX13 to adjust the lower and upper limits on a variable using sensitivity analysis information together with the best upper bound.
BINVRT	Basis inversion or reinversion is accomplished by this subroutine. The basis matrix consists of those columns P_j from Figure 3 for the basic variables j (specified in array IBV). The corresponding right-hand-side vector is $$RHS - \sum_j U_j \cdot P_j$$

Subroutine	Description
	where the summation is taken over all nonbasic variables j which are at upper bound (specified in arrays NBV and IUPPER). Subroutine INVERT is called to determine the basis inverse matrix and the corresponding values of the basic variables.
GETCØL	This subroutine places the J-th column from the tableau shown in Figure 3 into the array PJ which is assumed to have been set to zero prior to calling the subroutine. The number of nonzero entries in the column and indicators as to which entries are nonzero are returned using parameter NZERØS and array IS.
INPUT1	The number of less than or equal, equality, and greater than or equal constraints are read and the parameters in CØMMØN/P2/ are initialized. From the number of nonzero entries by column (array NZ), the column pointers are developed (array NP).
INPUT2	The arrays IR and IA associated with the constraint matrix representation are read, along with the number of entries in the table of constants (parameter NTC).
INPUT3	The remainder of the data associated with the constraint matrix, the table of constants and the right-hand-side vector, are read. The lower and upper limits on variables are read in when values other than the default values of 0 and $\div \infty$ (respectively) are desired. The cost data and lists of integer and/or concave variables are read in next. For a mixed integer linear program (ITYPE = 1), a determination is made if the objective function is integer valued. This information is used in subroutine BØX17 to round lower bounds on the nodes.
INPUT4	Called by subroutine BØX2, this subroutine establishes the initial basis, basis inverse and right-hand-side for the first subprogram. If the initial feasible basis is a part of the program input (IFB = 1 on the first control card), then this basis is read in and the basis inversion feature is used to establish the corresponding basis inverse and right-hand-side. Otherwise, the initial feasible basis consists of slack and arti-

Subroutine	Description
	ficial variables with the basis inverse being an identity matrix and the initial right-hand-side agrees with the input right-hand-side vector. In either case, if there are artificals in the initial basis, subroutine ØBJ1 is called to establish the phase 1 objective function.
INPUT5	This subroutine selects the linear programming algorithm (primal or dual simplex) to be used to solve the first subprogram. The initial basis is tested for primal feasibility (nonnegative right-hand-side) and, if it is primal feasible, the primal algorithm is selected. If the initial basis is not primal feasible, it is tested for dual feasibility (nonnegative reduced cost vector) and, if it is dual feasible, the dual simplex algorithm is selected.
INVERT	The Gauss-Jordan method of matrix inversion is used to invert the basis matrix and determine the corresponding vector of values of basic variables. An ill-conditioned basis matrix, if encountered, results in the termination of computations.
ØBJ1	This subroutine computes and stores the (linear programming) phase 1 objective function associated with the tableau for the first subprogram in its canonical form.
RSTART	The manipulation of job restart tapes (tape units 7 through 10) is done in this subroutine.

If a job is to commence from previously prepared restart tapes (indicated by the setting MSTART = 1 on the second control card), subroutine ICØN calls subroutine RSTART with IENTRY = 0. Tape units 7 and 8 are read, restoring the branch-and-bound list, labeled commons and variable dimensioned arrays to their condition at the time computations were last terminated. In subroutine ICØN, the flow of the branch-and-bound computation is then reentered at the point where computations were terminated.

The determination that job restart tapes are to be created is made in subroutine ICØN when the elapsed execution time exceeds the input maximum execution

Subroutine | Description

time (parameter TIME1 on the third control card). The current values of labeled commons (CØMMØN/P2/, /P3/ and /P4/) and variable dimensioned arrays are saved on tape unit 9. The branch-and-bound list together with intermediate data on tape unit 4 (if any) are saved on tape unit 10. Note that the data saved on tape unit 9 corresponds to the data read from tape unit 7 on a subsequent restart run. Similarly, tape unit 10 corresponds to tape unit 8.

SIMPLE

This subroutine contains the primal and dual simplex algorithms. Basis reinversion is used as a technique for suppressing round-off error in the simplex computations. The frequency of basis reinversion is specified by parameter MBINV on the first control card. In the primal algorithm, the entering variable is taken as the first nonbasic variable encountered having negative reduced cost. As compared with the usual rule in which all reduced costs are computed with the entering variable corresponding to the minimum reduced cost, this frequently results in less total computation in reaching the optimum. A common pivot logic is used for both the primal and dual simplex algorithms. Note that the phase 1 objective function is deleted when no artificials remain in the basis.

The computations in this subroutine can terminate when any one of the following determinations is made:

(i) The current solution is optimal;

(ii) The primal program is infeasible;

(iii) The primal program is unbounded; or

(iv) The dual value exceeds the best upper bound (for the dual simplex algorithm).

Unboundedness of the primal program is impossible since the lower and upper bounds define a compact set; however, this form of termination can occur due to the accumulation of round-off errors. Because the dual value in the dual simplex algorithm is monotone nondecreasing, comparison against the

Subroutine	Description
	best upper bound is a useful way of saving unnecessary computation.
SLØPES	This subroutine computes the sensitivity analysis slopes which are required by the branch-and-bound algorithm for penalty analyses. These are computed from the optimal tableau using the dual simplex entering variable selection criteria.
TIMEC	A printout of the elapsed job execution time is obtained by calling this subroutine.

There are six CDC 6700 system subroutines which are used in conjunction with the ICØN algorithm. Only three of these (ØPENMS, READMS and WRITMS) are nonstandard. These are described in Appendix C. Figure 5 shows the number of times each system routine is called by some subroutine in the algorithm.

	CALLING SUBROUTINE							
SUBROUTINE CALLED	MAIN	ICØN	BØX1	BØX12	BØX25	INVERT	RSTART	TIMEC
ØPENMS		2						
READMS				2			2	
WRITMS					2		2	
SECØND		1	1					1
EØF			1					
EXIT	1		3			1	1	

FIGURE 5

CALLS TO SYSTEM SUBROUTINES

APPENDIX F

LISTING OF THE ICØN ALGORITHM

This appendix presents a listing of the 26 basic subroutines which comprise the ICØN branch-and-bound algorithm. Not shown in Exhibit 4 are the three user provided subroutines (MAIN, READIN and GETØBJ) discussed in Appendix B, Volume 1.

EXHIBIT 4

LISTING OF THE ICØN ALGORITHM

```
      SUBROUTINE ICON (IF,F,NI,NF)
C BRANCH-AND-BOUND ALGORITHM FOR THE INTEGER CONCAVE PROGRAM.
      COMMON/PC/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,
     1          NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,
     2          NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,
     3          NF23,NF24,NFMS3
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      COMMON/A0/IMASS2(1001),IMASS3(1001)
      DIMENSION IF(NI),F(NF)
      EPSI=1.0E-11
      EPSIM=-EPSI
      BIGN=1.0E+100
      CALL OPENMS (2,IMASS2,1001,0)
      CALL OPENMS (3,IMASS3,1001,0)
  100 CALL BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,ND10,
     1           ND11,NDMS2,NDMS3)
      IF(MSTART.EQ.0)GOTO110
      CALL RSTART (IF,IF(NI8),IF(NI10),IF(NI11),IF(NI12),IF(NIMS2),
     1             F,F(NF6),F(NF7),F(NF24),F(NFMS3),NI,NF,ND4,ND5,
     2             ND10,NDMS2,NDMS3,0)
      GOTO170
  110 CALL BOX2 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),IF(NI5),IF(NI6),
     1           IF(NI7),IF(NI8),IF(NI9),IF(NI10),IF(NI11),IF(NI12),
     2           F(NF1),F(NF2),F(NF3),F(NF4),F(NF5),F(NF6),F(NF7),
     3           F(NF8),F(NF9),F(NF10),F(NF11),F(NF12),F(NF15),F(NF16),
     4           F(NF17),F(NF18),F(NF19),F(NF20),F(NF24),ND1,ND2,ND3,
     5           ND4,ND5,ND6,ND7,ND8,ND10)
      LNODE=2
      GOTO190
  120 IF(NLIST.NE.0)GOTO130
      CALL BOX5 (IF(NI8),F(NF12),F(NF13),F(NF14),ND1,ND10,ND11,0)
      GOTO100
  130 IF(IBUBOP.EQ.0)GOTO160
      CALL BOX7 (IF(NI8),F(NF13),ND10)
      IF(BLB.LT.UNOT)GOTO140
      UNOT=BLB + PCBUB
      IF(IOUTPT.NE.0)WRITE(6,1000)UNOT
  140 CALL BOX10 (IF(NI8),F(NF13),F(NF14),ND10,ND11)
  150 IF(NLISTS.EQ.0)GOTO120
  160 CALL BOX12 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),
     1            F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,
     2            ND9,ND10,ND11,NDMS2,NDMS3)
      CALL BOX13 (IF(NI5),IF(NI6),IF(NI10),F(NF4),F(NF15),F(NF16),
     1            F(NF18),F(NF19),F(NF20),F(NF21),F(NF22),ND1,ND4,ND6,
```

EXHIBIT 4 (Continued)

```
     2           ND8)                                                   ICON0054
      LNODE=1                                                           ICON0055
  170 CALL SECOND (TIME2)                                               ICON0056
      IF(TIME2.LT.BEGTM+TIME1)GOTO180                                   ICON0057
      CALL BOX5 (IF(NI8),F(NF12),F(NF13),F(NF14),ND1,ND10,ND11,1)       ICON0058
      CALL RSTART (IF,IF(NI8),IF(NI10),IF(NI11),IF(NI12),IF(NIMS2),     ICON0059
     1            F,F(NF6),F(NF7),F(NF24),F(NFMS3),NI,NF,ND4,ND5,       ICON0060
     2            ND10,NDMS2,NDMS3,1)                                   ICON0061
      GOTO100                                                           ICON0062
  180 CALL BOX15 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),IF(NI5),IF(NI6),      ICON0063
     1           IF(NI7),IF(NI9),IF(NI10),IF(NI11),IF(NI12),F(NF1),     ICON0064
     2           F(NF2),F(NF3),F(NF4),F(NF5),F(NF6),F(NF7),F(NF8),      ICON0065
     3           F(NF9),F(NF10),F(NF11),F(NF15),F(NF16),F(NF17),        ICON0066
     4           F(NF18),F(NF19),F(NF20),F(NF21),F(NF23),F(NF24),       ICON0067
     5           ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9)                   ICON0068
  190 IF(IEOJ.NE.0)GOTO250                                              ICON0069
      CALL BOX17 (IF(NI5),IF(NI6),IF(NI10),F(NF4),F(NF15),F(NF16),      ICON0070
     1           F(NF18),F(NF19),F(NF20),ND1,ND4,ND6,ND8)               ICON0071
      IF(IBUBOP.EQ.1)GOTO210                                            ICON0072
      IF(BOUNDL.LT.(1.0-TOL1)*UNOT)GOTO200                              ICON0073
      IF(IOUTPT.EQ.0)GOTO250                                            ICON0074
      WRITE(6,1001)                                                     ICON0075
      GOTO250                                                           ICON0076
  200 IF(IFEAS.EQ.0)GOTO240                                             ICON0077
      IF(BOUNDU.GE.UNOT)GOTO230                                         ICON0078
      GOTO220                                                           ICON0079
  210 IF(IFEAS.EQ.0)GOTO240                                             ICON0080
  220 CALL BOX23 (IF(NI8),F(NF12),F(NF13),F(NF18),ND1,ND6,ND10)         ICON0081
  230 IF(BOUNDU-BOUNDL.GT.TOL1*ABS(BOUNDU))GOTO240                      ICON0082
      IF(IOUTPT.EQ.0 .OR. ITYPE.EQ.1)GOTO250                            ICON0083
      WRITE(6,1002)                                                     ICON0084
      GOTO250                                                           ICON0085
  240 CALL BOX25 (IF(NI8),IF(NIMS2),F(NF4),F(NF13),F(NF14),F(NF15),     ICON0086
     1           F(NF16),F(NF21),F(NF22),F(NF23),F(NFMS3),ND1,ND6,      ICON0087
     2           ND9,ND10,ND11,NDMS2,NDMS3)                             ICON0088
  250 IF(IOUTPT.NE.0)CALL TIMEC                                         ICON0089
      IF(LNODE.NE.2)GOTO260                                             ICON0090
      IF(IBUBOP.EQ.1)GOTO150                                            ICON0091
      GOTO120                                                           ICON0092
  260 LNODE=2                                                           ICON0093
      GOTO170                                                           ICON0094
 1000 FORMAT(31H0THE PHASE 1 BEST UPPER BOUND =,E15.6)                  ICON0095
 1001 FORMAT(46H0THE LOWER BOUND EXCEEDS THE BEST UPPER BOUND.)         ICON0096
 1002 FORMAT(38H0THE LOWER AND UPPER BOUNDS ARE EQUAL.)                 ICON0097
      END                                                               ICON0098
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX1 (IF,F,NI,NF,ND1,ND2,ND3,ND4,ND5,ND6,ND7,ND8,ND9,
     1                 ND10,ND11,NDMS2,NDMS3)
C READ BRANCH-AND-BOUND INPUT. READ USER INPUT.
      COMMON/P0/NI1,NI2,NI3,NI4,NI5,NI6,NI7,NI8,NI9,NI10,NI11,NI12,
     1          NIMS2,NF1,NF2,NF3,NF4,NF5,NF6,NF7,NF8,NF9,NF10,NF11,
     2          NF12,NF13,NF14,NF15,NF16,NF17,NF18,NF19,NF20,NF21,NF22,
     3          NF23,NF24,NFMS3
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      DIMENSION IF(NI),F(NF)
      CALL SECOND (BEGTM)
C READ THE CONTROL PARAMETERS.
      READ(5,1000)N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,
     1            NBVRL2,NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,
     2            MBINV
      IF(EOF(5).NE.0.0)CALL EXIT
      IF(N.LE.0)CALL EXIT
      READ(5,1000)IOUTPT,ITRACE,MSTART
      IF(MXLIST.EQ.0)MXLIST=1000
      IF(MXLIST.GT.1000)MXLIST=1000
      IF(ITAPE.EQ.0)ITAPE=5
      IF(MXITER.EQ.0)MXITER=1000
      INDEX=0
      IF(MBINV.EQ.0 .AND. IFB.EQ.0 .AND. LISTOP.EQ.0)INDEX=1
      READ(5,1001)TIME1,TOL1,TOL2,UNOT,PCBUB
      IF(TIME1.EQ.0.0)TIME1=180.
      IF(TOL1.EQ.0.0)TOL1=EPSI
      IF(TOL2.EQ.0.0)TOL2=EPSI
      IF(UNOT.EQ.0.0 .OR. PCBUB.NE.0.0)UNOT=BIGN
      IBUBOP=0
      IF(PCBUB.NE.0.0)IBUBOP=1
      READ(5,1002)ALPHA
      WRITE(6,1003)N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,
     1             NBVRL2,NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,
     2             MBINV,IOUTPT,ITRACE,MSTART
      WRITE(6,1004)TIME1,TOL1,TOL2,UNOT,PCBUB
      WRITE(6,1005)ALPHA
      IF(ITRACE.GE.1)WRITE(6,1009)
      ND1=N
      NI1=1
      NI2=NI1 + ND1
      NI3=NI2 + ND1
C READ THE NUMBER OF CONSTRAINTS, THE NUMBER OF NONZERO ENTRIES IN THE
C CONSTRAINT MATRIX (BY COLUMN), AND DEVELOP THE COLUMN POINTERS.
      CALL INPUT1 (IF(NI1),IF(NI2),ND1)
      ND2=NSUM
      NI4=NI3 + ND2
      NI5=NI4 + ND2
C READ THE CONSTRAINT MATRIX COLUMN-BY-COLUMN.
```

EXHIBIT 4 (Continued)

```
      CALL INPUT2 (IF(NI1),IF(NI2),IF(NI3),IF(NI4),ND1,ND2)            BOX10058
      ND3=NTC                                                          BOX10059
      ND4=M + 2                                                        BOX10060
      ND5=N + M3                                                       BOX10061
      ND6=N + M3 + M + 2                                               BOX10062
      IF(ND6.LT.11)ND6=11                                              BOX10063
      ND7=ND4                                                          BOX10064
      IF(INDEX.EQ.1)ND7=1                                              BOX10065
      ND8=ND4                                                          BOX10066
      IF(NBVRL1.LE.2 .OR. NTITE1.EQ.0)GOTO110                          BOX10067
      IF(NSTRAT.EQ.1)GOTO100                                           BOX10068
      IF(NBVRL2.LE.2 .OR. NTITE2.EQ.0)GOTO110                          BOX10069
  100 ND9=1                                                            BOX10070
  110 ND9=ND1                                                          BOX10071
      IF(ITYPE.EQ.1 .OR. ITYPE.EQ.3)ND9=1                              BOX10072
      ND10=MXLIST                                                      BOX10073
      ND11=ND10                                                        BOX10074
      IF(NODRL1.EQ.1)GOTO130                                           BOX10075
      IF(NSTRAT.EQ.1)GOTO120                                           BOX10076
      IF(NODRL2.EQ.1)GOTO130                                           BOX10077
  120 ND11=1                                                           BOX10078
  130 NI6=NI5 + ND1                                                    BOX10079
      NI7=NI6 + ND1                                                    BOX10080
      NI8=NI7 + ND4                                                    BOX10081
      NI9=NI8 + ND10                                                   BOX10082
      NI10=NI9 + ND6                                                   BOX10083
      NI11=NI10 + ND4                                                  BOX10084
      NI12=NI11 + ND5                                                  BOX10085
      NITOT=NI12 + ND5                                                 BOX10086
      NF1=1                                                            BOX10087
      NF2=NF1 + ND3                                                    BOX10088
      NF3=NF2 + ND4                                                    BOX10089
      NF4=NF3 + ND4                                                    BOX10090
      NF5=NF4 + ND1                                                    BOX10091
      NF6=NF5 + ND5                                                    BOX10092
      NF7=NF6 + ND4                                                    BOX10093
      NF8=NF7 + ND5                                                    BOX10094
      NF9=NF8 + ND6                                                    BOX10095
      NF10=NF9 + ND4                                                   BOX10096
      NF11=NF10 + ND7*ND7                                              BOX10097
      NF12=NF11 + ND4                                                  BOX10098
      NF13=NF12 + ND1                                                  BOX10099
      NF14=NF13 + ND10                                                 BOX10100
      NF15=NF14 + ND11                                                 BOX10101
      NF16=NF15 + ND6                                                  BOX10102
      NF17=NF16 + ND6                                                  BOX10103
      NF18=NF17 + ND6                                                  BOX10104
      NF19=NF18 + ND6                                                  BOX10105
      NF20=NF19 + ND8                                                  BOX10106
      NF21=NF20 + ND8                                                  BOX10107
      NF22=NF21 + ND6                                                  BOX10108
      NF23=NF22 + ND6                                                  BOX10109
      NF24=NF23 + ND9                                                  BOX10110
      NFTOT=NF24 + ND4*ND4                                             BOX10111
      IF(NI.LT.NITOT .OR. NF.LT.NFTOT)GOTO230                          BOX10112
C READ THE TABLE OF CONSTANTS, THE RIGHT-HAND-SIDE, THE LOWER AND      BOX10113
C UPPER BOUNDS, THE COST DATA, AND THE LISTS CF INTEGER AND CONCAVE    BOX10114
```

```
C VARIABLES.                                                            BOX10115
      CALL INPUT3 (IF(NI5),IF(NI6),IF(NI9),F(NF1),F(NF2),F(NF4),        BOX10116
     1            F(NF15),F(NF16),F(NF17),ND1,ND3,ND4,ND6)              BOX10117
      WRITE(6,1006)NITOT,NFTOT                                          BOX10118
C ESTABLISH THE STRUCTURE OF THE BRANCH-AND-BOUND LIST.                 BOX10119
      IF(NSTRAT.EQ.2)GOTO140                                            BOX10120
      NIMS2=NI10 - 9                                                    BOX10121
      NDMS2=ND4 + 2*ND5 + 9                                             BOX10122
      GOTO150                                                           BOX10123
  140 NIMS2=NI10 - 11                                                   BOX10124
      NDMS2=ND4 + 2*ND5 + 11                                            BOX10125
  150 NDMS3=2*ND6 + 3                                                   BOX10126
      IF(NTITE1.EQ.0)GOTO160                                            BOX10127
      IF(NSTRAT.EQ.1)GOTO170                                            BOX10128
      IF(NTITE2.EQ.1)GOTO170                                            BOX10129
  160 NFMS3=NF18 - 3                                                    BOX10130
      NDMS3=NDMS3 + ND6 + 2*ND8                                         BOX10131
      GOTO180                                                           BOX10132
  170 NFMS3=NF21 - 3                                                    BOX10133
  180 IF(NSTRAT.EQ.1)GOTO190                                            BOX10134
      NFMS3=NFMS3 - 1                                                   BOX10135
      NDMS3=NDMS3 + 1                                                   BOX10136
  190 IF(LISTOP.EQ.1)GOTO200                                            BOX10137
      NDMS3=NDMS3 + ND9 + ND4*ND4                                       BOX10138
      GOTO210                                                           BOX10139
  200 IF(ITYPE.EQ.1)GOTO210                                             BOX10140
      NDMS3=NDMS3 + ND9                                                 BOX10141
  210 NMSTOT=NDMS2 + NDMS3                                              BOX10142
      WRITE(6,1007)NMSTOT,NDMS2,NDMS3                                   BOX10143
      IF(ITYPE.EQ.1)GOTO220                                             BOX10144
C READ USER INPUT.                                                      BOX10145
      CALL READIN                                                       BOX10146
  220 CALL TIMEC                                                        BOX10147
      RETURN                                                            BOX10148
  230 WRITE(6,1006)NITOT,NFTOT                                          BOX10149
      WRITE(6,1008)NI,NF                                                BOX10150
      CALL TIMEC                                                        BOX10151
      CALL EXIT                                                         BOX10152
      RETURN                                                            BOX10153
 1000 FORMAT(16I5)                                                      BOX10154
 1001 FORMAT(5E12.0)                                                    BOX10155
 1002 FORMAT(10A8)                                                      BOX10156
 1003 FORMAT(37H1INPUT FOR BRANCH-AND-BOUND ALGORITHM/                  BOX10157
     1      21H0INTEGER PARAMETERS =,16I7/21X,3I7)                      BOX10158
 1004 FORMAT(21H0REAL    PARAMETERS =,5E15.6)                           BOX10159
 1005 FORMAT(21H0PROGRAM IDENTIFIER =,4X,10A8)                          BOX10160
 1006 FORMAT(44H0ADEQUATE DIMENSIONS FOR ARRAYS IF AND F ARE,I10,       BOX10161
     1      4H AND,I10,16H (RESPECTIVELY).)                             BOX10162
 1007 FORMAT(39H0THE BRANCH-AND-BOUND LIST CONSISTS OF ,I10,21H LOCATIONBOX10163
     1S PER NODE (,I10,25H LOCATIONS ON UNIT 2 AND ,I10,22H LOCATIONS ONBOX10164
     2 UNIT 3).)                                                        BOX10165
 1008 FORMAT(15H0THE DIMENSIONS,I10,4H AND,I10,18H ARE INSUFFICIENT.)   BOX10166
 1009 FORMAT(10H *****BOX1)                                             BOX10167
      END                                                               BOX10168
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX2 (NZ,NP,IR,IA,INT,ICC,IS,INUSE,NV,IBV,NBV,IUPPER,
     1                 TC,BORIG,RHS,C2,C1,BI,BN,U,PJ,BINV,XJ,XNOT,
     2                 SIGMAL,SIGMAU,V,XZ,S0,S1,B,ND1,ND2,ND3,ND4,ND5,
     3                 ND6,ND7,ND8,ND10)
C SOLVE THE FIRST SUBPROGRAM.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      DIMENSION INT(ND1),ICC(ND1),IS(ND4),INUSE(ND10)
      DIMENSION BORIG(ND4),RHS(ND4),C2(ND1),U(ND6),PJ(ND4),XNOT(ND1),
     1          SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),S0(ND8),S1(ND8)
      IF(ITRACE.GE.1)WRITE(6,1007)
C INITIALIZE THE BEST UPPER BOUND AND LIST.
      LPHASE=1
      NODNOT=0
      DO100J=1,N
  100 XNOT(J)=0.0
      NLIST=0
      NLISTS=0
      NFEAS=0
      LSTMX=0
      ITRTOT=0
      ITRMAX=0
      DO110I=1,MXLIST
  110 INUSE(I)=0
      NODRUL=NODRL1
      NBVRUL=NBVRL1
      NTIGHT=NTITE1
      PBRNOD=1.0
      NUPDWN=1
      NODE=1
      NBRNOD=0
      IF(IOUTPT.NE.0)WRITE(6,1000)NODE,NBRNOD
C ESTABLISH THE UPPER BOUNDS FOR THE FIRST SUBPROGRAM.
      DO120J=1,N1P2
  120 U(J)=SIGMAU(J)-SIGMAL(J)
C ESTABLISH THE COST DATA FOR THE FIRST SUBPROGRAM.
      TSIG=0.0
      DO150J=1,N
      K=ICC(J)
      IF(K.EQ.0)GOTO140
      CALL GETOBJ (K,SIGMAL(J),F0)
      TSIG=TSIG + F0
      IF(ABS(U(J)).LE.TOL2)GOTO130
      CALL GETOBJ (K,SIGMAU(J),F1)
      C2(J)=(F1-F0)/U(J)
      GOTO150
  130 C2(J)=0.0
      GOTO150
  140 TSIG=TSIG + C2(J)*SIGMAL(J)
```

EXHIBIT 4 (Continued)

```
  150 CONTINUE                                                            BOX20058
      IF(IOUTPT.LE.1)GOTO190                                              BOX20059
      WRITE(6,1001)TSIG                                                   BOX20060
      WRITE(6,1002)                                                       BOX20061
      DO170J=1,N                                                          BOX20062
      K=INT(J)                                                            BOX20063
      L=ICC(J)                                                            BOX20064
      IF(IOUTPT.GE.3)GOTO160                                              BOX20065
      IF(K.EQ.0 .AND. L.EQ.0)GOTO170                                      BOX20066
  160 WRITE(6,1003)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)                        BOX20067
  170 CONTINUE                                                            BOX20068
      IF(IOUTPT.LE.2)GOTO190                                              BOX20069
      IF(NM1M3.EQ.N)GOTO190                                               BOX20070
      DO180J=NP1,NM1M3                                                    BOX20071
  180 WRITE(6,1004)J,SIGMAL(J),SIGMAU(J)                                  BOX20072
  190 CONTINUE                                                            BOX20073
C ESTABLISH THE RIGHT-HAND-SIDE FOR THE FIRST SUBPROGRAM.                 BOX20074
      DO200I=1,MP1                                                        BOX20075
  200 RHS(I)=BORIG(I)                                                     BOX20076
      IPHASE=1                                                            BOX20077
      DO230J=1,N                                                          BOX20078
      IF(ABS(SIGMAL(J)).LE.EPSI)GOTO230                                   BOX20079
      DO210I=1,MP1                                                        BOX20080
  210 PJ(I)=0.0                                                           BOX20081
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,    BOX20082
     1            J,NZEROS)                                               BOX20083
      DO220I1=1,NZEROS                                                    BOX20084
      I=IS(I1)                                                            BOX20085
  220 RHS(I)=RHS(I) - PJ(I)*SIGMAL(J)                                     BOX20086
  230 CONTINUE                                                            BOX20087
C ESTABLISH THE INITIAL BASIS, BASIS INVERSE, AND VALUES OF THE BASIC     BOX20088
C VARIABLES FOR THE FIRST SUBPROGRAM.                                     BOX20089
      CALL INPUT4 (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,BORIG,RHS,C2,C1,   BOX20090
     1            BI,BN,U,PJ,BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)          BOX20091
C DETERMINE THE APPLICABLE LP ALGORITHM.                                  BOX20092
      CALL INPUT5 (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,B,BOX20093
     1            ND1,ND2,ND3,ND4,ND5,ND6)                                BOX20094
      IF(IALGO.NE.0)GOTO240                                               BOX20095
      IEOJ=1                                                              BOX20096
      RETURN                                                              BOX20097
C SOLVE THE FIRST SUBPROGRAM.                                             BOX20098
  240 CALL SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,BN,   BOX20099
     1            U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)        BOX20100
      IF(IEOJ.NE.0)RETURN                                                 BOX20101
      NFEAS=NFEAS + 1                                                     BOX20102
C PRINT THE SOLUTION.                                                     BOX20103
      Z=Z+TSIG                                                            BOX20104
      DO250J=1,N1P2                                                       BOX20105
  250 XZ(J)=XZ(J) + SIGMAL(J)                                             BOX20106
      IF(IOUTPT.EQ.0)GOTO260                                              BOX20107
      NUP=N                                                               BOX20108
      IF(IOUTPT.GE.3)NUP=NM1M3                                            BOX20109
      WRITE(6,1005)Z                                                      BOX20110
      WRITE(6,1006)(XZ(J),J=1,NUP)                                        BOX20111
  260 IF(ND8.EQ.1)RETURN                                                  BOX20112
      IF(NBVRUL.LE.2 .OR. NTIGHT.EQ.0)GOTO280                             BOX20113
C INITIALIZE THE SLOPES.                                                  BOX20114
```

EXHIBIT 4 (Continued)

```
      DO270I=1,M7                                                       BOX20115
      S0(I)=0.0                                                         BOX20116
  270 S1(I)=0.0                                                         BOX20117
      RETURN                                                            BOX20118
C DETERMINE THE SLOPES ASSOCIATED WITH THE OPTIMAL OBJECTIVE VALUE.      BOX20119
  280 CALL SLOPES (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,PJ,XJ,S0, BOX20120
     1           S1,B,ND1,ND2,ND3,ND4,ND5,ND8)                          BOX20121
      RETURN                                                            BOX20122
 1000 FORMAT(1H1,50(1H*)/6H0NODE ,I5/20H0BRANCHED FROM NODE ,I5)        BOX20123
 1001 FORMAT(7H0TSIG =,E15.6)                                           BOX20124
 1002 FORMAT(9H0VARIABLE,4X,3HINT,8X,2HCC,12X,5HLOWER,12X,5HUPPER,13X,  BOX20125
     1       4HCOST/2X,6HNUMBER,3X,8HVARIABLE,2X,8HVARIABLE,9X,5HLIMIT, BOX20126
     2       12X,5HLIMIT,9X,11HCOEFFICIENT/12X,6HNUMBER,4X,6HNUMBER//)  BOX20127
 1003 FORMAT(3X,I5,2(5X,I5),3X,3(E15.6,2X))                             BOX20128
 1004 FORMAT(3X,I5,23X,E15.6,2X,E15.6)                                  BOX20129
 1005 FORMAT(17H0SOLUTION VALUE =,E15.6)                                BOX20130
 1006 FORMAT(17H0VARIABLES       =,6E15.6/(17X,6E15.6))                 BOX20131
 1007 FORMAT(10H *****BOX2)                                             BOX20132
      END                                                               BOX20133
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX5 (INUSE,XNOT,CAPP,CAPL,ND1,ND10,ND11,IENTRY)
C PRINT THE SOLUTICN.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IF8,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BCUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION INUSE(ND10)
      DIMENSION XNOT(ND1),CAPP(ND10),CAPL(ND11)
      IF(ITRACE.GE.1)WRITE(6,1017)
C PRINT OUT PROGRAM IDENTIFICATION.
      WRITE(6,1000)N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,
     1               NBVRL2,NTITE2,MXLIST,LISTOP,ITAPE,IF8,MXITER,
     2               MBINV,IOUTPT,ITRACE,MSTART
      WRITE(6,1001)TIME1,TOL1,TOL2,UNOT,PCBUB
      WRITE(6,1002)ALPHA
C PRINT OUT THE SOLUTION OR THE CURRENT BEST SOLUTION.
      IF(IENTRY.EQ.1)GOTO100
      WRITE(6,1003)NODNOT
      REWIND 9
      REWIND 10
      WRITE(9)NODNOT
      WRITE(10)NODNOT
      END FILE 9
      END FILE 10
      GOTO110
  100 WRITE(6,1004)NODNOT
  110 WRITE(6,1005)UNOT
      WRITE(6,1006)(XNOT(I),I=1,N)
      WRITE(6,1007)NODE
      WRITE(6,1008)NFEAS
      WRITE(6,1009)LSTMX
      WRITE(6,1010)ITRTCT
      WRITE(6,1011)ITRMAX
      IF(IENTRY.NE.1)GOTO180
C PRINT OUT THE LIST.
      WRITE(6,1012)NLIST
      INDEX=0
      IF(NSTRAT.EQ.2)GOTO120
      IF(NODRL1.EQ.1)GOTO140
      GOTO130
  120 IF(NODRL2.EQ.1)GOTO140
      IF(LPHASE.EQ.1 .AND. NODRL1.EQ.1)GOTO140
  130 INDEX=1
      WRITE(6,1013)
      GOTO150
  140 WRITE(6,1014)
  150 DO170I=1,MXLIST
      IF(INUSE(I).EQ.0)GOTO170
      IF(INDEX.EQ.0)GOTO160
      WRITE(6,1015)INUSE(I),CAPP(I)
      GOTO170
  160 WRITE(6,1016)INUSE(I),CAPP(I),CAPL(I)
  170 CONTINUE
```

EXHIBIT 4 (Continued)

```
 180 CALL TIMEC                                                       BOX50058
     RETURN                                                           BOX50059
1000 FORMAT(21H1INTEGER PARAMETERS =,1(I7/21X,3I7)                    BOX50060
1001 FORMAT(21H0REAL    PARAMETERS =,5E15.6)                          BOX50061
1002 FORMAT(21H0PROGRAM IDENTIFIER =,4X,10A8)                         BOX50062
1003 FORMAT(66H0THE SOLUTION TO THE INTEGER CONCAVE PROGRAM WAS PROVIDEBOX50063
    1D BY NODE ,I5)                                                   BOX50064
1004 FORMAT(79H0THE CURRENT BEST SOLUTION TO THE INTEGER CONCAVE PROGRABOX50065
    1M WAS PROVIDED BY NODE ,I5)                                      BOX50066
1005 FORMAT(17H0SOLUTION VALUE =,E15.6)                               BOX50067
1006 FORMAT(17H0VARIABLES      =,6E15.6/(17X,6E15.6))                 BOX50068
1007 FORMAT(34H0THE NUMBER OF NODES EXAMINED WAS ,I5)                 BOX50069
1008 FORMAT(32H THE NUMBER OF NODES SOLVED WAS ,2X,I5)                BOX50070
1009 FORMAT(27H0THE MAXIMUM LIST SIZE WAS ,I5)                        BOX50071
1010 FORMAT(54H0THE TOTAL NUMBER OF SIMPLEX ITERATIONS PERFORMED WAS , BOX50072
    1   I7)                                                           BOX50073
1011 FORMAT(51H THE MAXIMUM NUMBER PERFORMED ALONG ANY BRANCH WAS ,   BOX50074
    1      3X,I7)                                                     BOX50075
1012 FORMAT(26H0THE CURRENT LIST SIZE IS ,I5)                         BOX50076
1013 FORMAT(1H0,3X,4HNODE,9X,5HLOWER/17X,5HBOUND//)                   BOX50077
1014 FORMAT(1H0,3X,4HNODE,9X,5HLOWER,8X,10HPROCESSING/17X,5HBOUND,    BOX50078
    1      11X,5HORDER//)                                             BOX50079
1015 FORMAT(3X,I5,2X,E15.6)                                           BOX50080
1016 FORMAT(3X,I5,2X,E15.6,2X,F11.0)                                  BOX50081
1017 FORMAT(10H *****BOX5)                                            BOX50082
     END                                                              BOX50083
```

```
      SUBROUTINE BOX7 (INUSE,CAPP,ND10)                                 BOX70001
C DETERMINE THE BEST LOWER BOUND.                                       BOX70002
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,    BOX70003
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFE,MXITER,MEINV,IOUTPT,     BOX70004
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)           BOX70005
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, BOX70006
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10                             BOX70007
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,   BOX70008
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,     BOX70009
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, BOX70010
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,   BOX70011
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ       BOX70012
      DIMENSION INUSE(ND10)                                             BOX70013
      DIMENSION CAPP(ND10)                                              BOX70014
      IF(ITRACE.GE.1)WRITE(6,1000)                                      BOX70015
      BLB=BIGN                                                          BOX70016
      DO100I=1,MXLIST                                                   BOX70017
      IF(INUSE(I).EQ.0)GOTO100                                          BOX70018
      IF(CAPP(I).GE.BLB)GOTO100                                         BOX70019
      BLB=CAPP(I)                                                       BOX70020
  100 CONTINUE                                                          BOX70021
      RETURN                                                            BOX70022
 1000 FORMAT(10H *****BOX7)                                             BOX70023
      END                                                               BOX70024
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX10 (INUSE,CAPP,CAPL,ND10,ND11)                      BOX10001
C SELECT A NODE FROM THE LIST AND PUT IT IN THE SUBLIST.                BOX10002
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,    BOX10003
     1           NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,    BOX10004
     2           ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)          BOX10005
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,   BOX10006
     1           NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,    BOX10007
     2           NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, BOX10008
     3           TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,  BOX10009
     4           L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ      BOX10010
      DIMENSION INUSE(ND10)                                             BOX10011
      DIMENSION CAPP(ND10),CAPL(ND11)                                   BOX10012
      IF(ITRACE.GE.1)WRITE(6,1000)                                      BOX10013
C SELECT THE NODE.                                                      BOX10014
      PBRNOD=0.0                                                        BOX10015
      DO100I=1,MXLIST                                                   BOX10016
      IF(INUSE(I).EQ.0)GOTO100                                          BOX10017
      IF(CAPP(I).GE.UNOT)GOTO100                                        BOX10018
      IF(CAPL(I).LE.PBRNOD)GOTO100                                      BOX10019
      PBRNOD=CAPL(I)                                                    BOX10020
      I0=I                                                              BOX10021
  100 CONTINUE                                                          BOX10022
C PUT IT IN THE SUBLIST.                                                BOX10023
      INUSE(I0)=-INUSE(I0)                                              BOX10024
      NLISTS=NLISTS+1                                                   BOX10025
      RETURN                                                            BOX10026
 1000 FORMAT(11H *****BOX10)                                            BOX10027
      END                                                               BOX10028
      SUBROUTINE BOX12 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,
     1                  SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NDMS2,
     2                  NDMS3)
C SELECT THE BRANCHING NODE FROM THE SUBLIST (PHASE 1) OR THE LIST
C (PHASE 2).
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,NBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,LNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,ELB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION INUSE(ND10),IMS(NDMS2)
      DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6),
     1          SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)
      IF(ITRACE.GE.1)WRITE(6,1000)
      IF(IBUBOP.EQ.0)GOTO110
C***********************************************************************
C SELECT THE BRANCHING NODE FROM THE SUBLIST.
C***********************************************************************
      PBRNOD=0.0
      DO100I=1,MXLIST
      IF(INUSE(I).GE.0)GOTO100
      IF(CAPL(I).LE.PBRNOD)GOTO100
      PBRNOD=CAPL(I)
      NBRNOD=-INUSE(I)
      I0=I
  100 CONTINUE
C DECREMENT THE SUBLIST COUNTER.
      NLISTS=NLISTS-1
      GOTO150
C***********************************************************************
C SELECT THE BRANCHING NODE FROM THE LIST.
C***********************************************************************
  110 IF(NODRUL.EQ.1)GOTO130
C PRIORITY NODE SELECTION RULE.
      BLB=BIGN
      DO120I=1,MXLIST
      IF(INUSE(I).EQ.0)GOTO120
      IF(CAPP(I).GE.BLB)GOTO120
      BLB=CAPP(I)
      NBRNOD=INUSE(I)
      I0=I
  120 CONTINUE
      GOTO150
C LIFO NODE SELECTION RULE.
  130 PBRNOD=0.0
      DO140I=1,MXLIST
      IF(INUSE(I).EQ.0)GOTO140
      IF(CAPL(I).LE.PBRNOD)GOTO140
      PBRNOD=CAPL(I)
      NBRNOD=INUSE(I)
      I0=I
  140 CONTINUE
```

EXHIBIT 4 (Continued)

```
C DECREMENT THE LIST COUNTER.
  150 INUSE(IO)=0
      NLIST=NLIST-1
C************************************************************************
C READ IN THE DATA FOR THE BRANCHING NODE.
C************************************************************************
      IF(NODE.EQ.NBRNCD)GOTO200
      CALL READMS (2,IMS,NDMS2,IO)
      CALL READMS (3,FMS,NDMS3,IO)
      IBRVR1=IMS(1)
      IUPDN1=IMS(2)
      L10=IMS(3)
      NITER=IMS(4)
      NBINV=IMS(5)
      M7=IMS(6)
      IPHASE=IMS(7)
      NPHASE=IMS(8)
      NM3M7=IMS(9)
      IF(NSTRAT.EQ.1)GOTO160
      IBRVR2=IMS(10)
      IUPDN2=IMS(11)
  160 Z=FMS(1)
      TSIG=FMS(2)
      XBRVR1=FMS(3)
      IF(NSTRAT.EQ.1)GOTO170
      XBRVR2=FMS(4)
  170 DO180J=1,N1P2
      SIGMAL(J)=SLOLD(J)
  180 SIGMAU(J)=SUOLD(J)
      IF(ITYPE.EQ.1)GOTO200
      DO190J=1,N
  190 C2(J)=C2OLD(J)
  200 TBRNOD=TSIG
C ADJUST THE BRANCHING VARIABLE SELECTION ACCORDING TO THE CURRENT
C PHASE OF THE ALGORITHM.
      IF(NSTRAT.EQ.1)GOTO210
      IF(LPHASE.EQ.1)GOTO210
      NBRVAR=IBRVR2
      NUPDWN=IUPDN2
      XBRNOD=XBRVR2
      RETURN
  210 NBRVAR=IBRVR1
      NUPDWN=IUPDN1
      XBRNOD=XBRVR1
      RETURN
 1000 FORMAT(11H *****BOX12)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX13 (INT,ICC,IBV,C2,SIGMAL,SIGMAU,XZ,S0,S1,SLOLD,
     1                  SUOLD,ND1,ND4,ND6,ND8)
C TIGHTEN THE LIMITS ON THE MONOTONE VARIABLES (PHASE 1) OR TIGHTEN
C THE LIMITS ON ALL VARIABLES USING THE BEST UPPER BOUND (PHASE 2).
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBLB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRICT,ITRMAX,BLE,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION INT(ND1),ICC(ND1),IBV(ND4)
      DIMENSION C2(ND1),SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),S0(ND8),S1(ND8),
     1          SLOLD(ND6),SUOLD(ND6)
      IF(ITRACE.GE.1)WRITE(6,1005)
      IF(NTIGHT.EQ.1)RETURN
      NCOUNT=0
      IF(UNOT.EQ.BIGN)GOTO310
      IF(IBUBOP.EQ.1)GOTO310
C***********************************************************************
C TIGHTEN THE LIMITS ON THE BASIC VARIABLES.
C***********************************************************************
      DO300I=1,M7
      IF(I.EQ.MP1)GOTO300
      J=IBV(I)
      IF(ABS(SIGMAU(J)-SIGMAL(J)).LE.TOL2)GOTO300
      XZJJJ=XZ(J)
      IF(XZJJJ.GT.SIGMAL(J)+TOL2)GOTO100
      S0(I)=-BIGN
      XZJJJ=SIGMAL(J)
      GOTO110
  100 IF(XZJJJ.LT.SIGMAU(J)-TOL2)GOTO110
      S1(I)=BIGN
      XZJJJ=SIGMAU(J)
  110 CONTINUE
C ALL VARIABLE TYPES.
      V1=SIGMAL(J)
      IF(S0(I).EQ.-BIGN)GOTO120
      T1=Z + S0(I)*(V1 - XZJJJ)
      GOTO130
  120 T1=BIGN
  130 IF(J.GT.N)GOTO230
      IF(INT(J).EQ.0)GOTO230
C INTEGER LINEAR OR INTEGER CONCAVE VARIABLE.
      K=XZJJJ
      V2=K
      V3=K+1
      K=ICC(J)
      IF(K.EQ.0)GOTO150
      CALL GETOBJ (K,V1,F1)
      CALL GETOBJ (K,V2,F2)
      CALL GETOBJ (K,V3,F3)
      IF(S0(I).EQ.-BIGN .AND. ABS(XZJJJ-V2).GT.TOL2)GOTO170
      IF(S0(I).EQ.-BIGN)GOTO140
      T2=Z + SQ(I)*(V2 - XZJJJ) + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))          BOX10058
      GOTO180                                                               BOX10059
  140 T2=Z + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))                               BOX10060
      GOTO180                                                               BOX10061
  150 IF(S0(I).EQ.-BIGN .AND. ABS(XZJJJ-V2).GT.TOL2)GOTO170                 BOX10062
      IF(S0(I).EQ.-BIGN)GOTO160                                             BOX10063
      T2=Z + S0(I)*(V2 - XZJJJ)                                             BOX10064
      GOTO180                                                               BOX10065
  160 T2=Z                                                                  BOX10066
      GOTO180                                                               BOX10067
  170 T2=BIGN                                                               BOX10068
  180 IF(S1(I).EQ.BIGN .AND. ABS(V3-XZJJJ).GT.TOL2)GOTO220                  BOX10069
      IF(K.EQ.0)GOTO200                                                     BOX10070
      IF(S1(I).EQ.BIGN)GOTO190                                              BOX10071
      T3=Z + S1(I)*(V3 - XZJJJ) + F3 - (F1 + C2(J)*(V3-SIGMAL(J)))          BOX10072
      GOTO260                                                               BOX10073
  190 T3=Z + F3 - (F1 + C2(J)*(V3-SIGMAL(J)))                               BOX10074
      GOTO260                                                               BOX10075
  200 IF(S1(I).EQ.BIGN)GOTO210                                              BOX10076
      T3=Z + S1(I)*(V3 - XZJJJ)                                             BOX10077
      GOTO260                                                               BOX10078
  210 T3=Z                                                                  BOX10079
      GOTO260                                                               BOX10080
  220 T3=BIGN                                                               BOX10081
      GOTO260                                                               BOX10082
C LINEAR OR CONCAVE VARIABLE.                                               BOX10083
  230 V2=XZJJJ                                                              BOX10084
      V3=V2                                                                 BOX10085
      IF(J.GT.N)GOTO240                                                     BOX10086
      K=ICC(J)                                                              BOX10087
      IF(K.EQ.0)GOTO240                                                     BOX10088
      CALL GETOBJ (K,V1,F1)                                                 BOX10089
      CALL GETOBJ (K,V2,F2)                                                 BOX10090
      T2=Z + F2 - (F1 + C2(J)*(V2-SIGMAL(J)))                               BOX10091
      GOTO250                                                               BOX10092
  240 T2=Z                                                                  BOX10093
  250 T3=T2                                                                 BOX10094
C ALL VARIABLE TYPES.                                                       BOX10095
  260 V4=SIGMAU(J)                                                          BOX10096
      IF(S1(I).EQ.BIGN)GOTO270                                              BOX10097
      T4=Z + S1(I)*(V4 - XZJJJ)                                             BOX10098
      GOTO280                                                               BOX10099
  270 T4=BIGN                                                               BOX10100
  280 CALL ADJUST (V1,V2,V3,V4,T1,T2,T3,T4,SL,SU)                           BOX10101
      SIGMAL(J)=SL                                                          BOX10102
      SIGMAU(J)=SU                                                          BOX10103
      IF(J.GT.N)GOTO290                                                     BOX10104
      IF(INT(J).EQ.0)GOTO290                                                BOX10105
C FOR INTEGER VARIABLES, CHECK THAT THE LOWER AND UPPER LIMITS ARE           BOX10106
C INTEGERS.                                                                 BOX10107
      K=SL                                                                  BOX10108
      SIGMAL(J)=K                                                           BOX10109
      IF(ABS(SL - SIGMAL(J)).GT.TOL2)SIGMAL(J)=K+1                          BOX10110
      K=SU                                                                  BOX10111
      SIGMAU(J)=K+1                                                         BOX10112
      IF(ABS(SU - SIGMAU(J)).GT.TOL2)SIGMAU(J)=K                            BOX10113
C INCREMENT THE COUNTER IF THE LIMITS HAVE CHANGED.                         BOX10114
```

EXHIBIT 4 (Continued)

```
  290 IF(ABS(SIGMAL(J)-SLOLD(J)).GT.TOL2)NCOUNT=NCOUNT+1                 BOX10115
      IF(ABS(SIGMAU(J)-SUOLD(J)).GT.TOL2)NCOUNT=NCOUNT+1                 BOX10116
  300 CONTINUE                                                           BOX10117
      GOTO370                                                            BOX10118
C***********************************************************************BOX10119
C ADJUST THE LIMITS ON THE MONOTONE VARIABLES.                           BOX10120
C***********************************************************************BOX10121
  310 DO360I=1,M7                                                        BOX10122
      IF(I.EQ.MP1)GOTO360                                                BOX10123
      J=IBV(I)                                                           BOX10124
      IF(ABS(SIGMAU(J)-SIGMAL(J)).LE.TOL2)GOTO360                        BOX10125
      XZJJJ=XZ(J)                                                        BOX10126
      IF(XZJJJ.LE.SIGMAL(J)+TOL2)GOTO330                                 BOX10127
      IF(S0(I).NE.-BIGN)GOTO330                                          BOX10128
C ADJUST THE LOWER LIMIT.                                                BOX10129
      IF(J.GT.N)GOTO320                                                  BOX10130
      IF(INT(J).EQ.0)GOTO320                                             BOX10131
      K=XZJJJ                                                            BOX10132
      SIGMAL(J)=K                                                        BOX10133
      IF(ABS(XZJJJ-SIGMAL(J)).GT.TOL2)SIGMAL(J)=K+1                      BOX10134
      GOTO330                                                            BOX10135
  320 SIGMAL(J)=XZJJJ                                                    BOX10136
  330 IF(XZJJJ.GE.SIGMAU(J)-TOL2)GOTO350                                 BOX10137
      IF(S1(I).NE.BIGN)GOTO350                                           BOX10138
C ADJUST THE UPPER LIMIT.                                                BOX10139
      IF(J.GT.N)GOTO340                                                  BOX10140
      IF(INT(J).EQ.0)GOTO340                                             BOX10141
      K=XZJJJ                                                            BOX10142
      SIGMAU(J)=K+1                                                      BOX10143
      IF(ABS(XZJJJ-SIGMAU(J)).GT.TOL2)SIGMAU(J)=K                        BOX10144
      GOTO350                                                            BOX10145
  340 SIGMAU(J)=XZJJJ                                                    BOX10146
C INCREMENT THE COUNTER IF THE LIMITS HAVE CHANGED.                      BOX10147
  350 IF(ABS(SIGMAL(J)-SLOLD(J)).GT.TOL2)NCOUNT=NCOUNT+1                 BOX10148
      IF(ABS(SIGMAU(J)-SUOLD(J)).GT.TOL2)NCOUNT=NCOUNT+1                 BOX10149
  360 CONTINUE                                                           BOX10150
C***********************************************************************BOX10151
C PRINT THE OLD AND NEW LIMITS.                                          BOX10152
C***********************************************************************BOX10153
  370 IF(IOUTPT.LE.2)RETURN                                              BOX10154
      IF(UNOT.EQ.BIGN)GOTO380                                            BOX10155
      IF(IBUBOP.EQ.1)GOTO380                                             BOX10156
      WRITE(6,1000)NBRNOD                                                BOX10157
      GOTO390                                                            BOX10158
  380 WRITE(6,1001)NBRNOD                                                BOX10159
  390 IF(NCOUNT.EQ.0)GOTO410                                             BOX10160
      WRITE(6,1002)NCOUNT                                                BOX10161
      DO400I=1,M7                                                        BOX10162
      IF(I.EQ.MP1)GOTO400                                                BOX10163
      J=IBV(I)                                                           BOX10164
      IF(ABS(SIGMAL(J)-SLOLD(J)).LE.TOL2 .AND.                           BOX10165
     1   ABS(SIGMAU(J)-SUOLD(J)).LE.TOL2)GOTO400                         BOX10166
      WRITE(6,1003)J,SLOLD(J),SIGMAL(J),XZ(J),SIGMAU(J),SUOLD(J)         BOX10167
  400 CONTINUE                                                           BOX10168
      GOTO420                                                            BOX10169
  410 WRITE(6,1004)                                                      BOX10170
  420 CALL TIMEC                                                         BOX10171
```

EXHIBIT 4 (Continued)

```
      RETURN                                                            BOX10172
 1000 FORMAT(1H0,50(1H*)/52H0TIGHTEN THE LIMITS ON THE BASIC VARIABLES FBOX10173
     1OR NODE ,I5)                                                      BOX10174
 1001 FORMAT(1H0,50(1H*)/54H0ADJUST THE LIMITS ON THE MONOTONE VARIABLESBOX10175
     1 FOR NODE ,I5)                                                    BOX10176
 1002 FORMAT(12H0THERE WERE ,I5,23H CHANGES TO THE LIMITS./             BOX10177
     1        1H0,2X,5HBASIC,11X,3HOLD,14X,3HNEW,11X,8HVARIABLE,12X,    BOX10178
     2        3HNEW,14X,3HOLD/1X,8HVARIABLE,2(9X,5HLOWER,3X),9X,        BOX10179
     3        5HVALUE,2(12X,5HUPPER)/6X,2(12X,5HLIMIT),17X,             BOX10180
     4        2(12X,5HLIMIT)//)                                         BOX10181
 1003 FORMAT(3X,I5,1X,5(2X,E15.6))                                      BOX10182
 1004 FORMAT(37H0THERE WERE NO CHANGES TO THE LIMITS.)                  BOX10183
 1005 FORMAT(11H *****BOX13)                                            BOX10184
      END                                                               BOX10185
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX15 (NZ,NP,IR,IA,INT,ICC,IS,AV,IBV,NBV,IUPPER,TC,    BOX10001
     1                  BORIG,RHS,C2,C1,BI,BN,U,PJ,BINV,XJ,SIGMAL,       BOX10002
     2                  SIGMAU,V,XZ,S0,S1,SLOLD,C2OLD,B,ND1,ND2,ND3,     BOX10003
     3                  ND4,ND5,ND6,ND7,ND8,ND9)                         BOX10004
C SOLVE SUBPROGRAM LNODE.                                               BOX10005
      COMMON/P1/N,M,ITYPE,NSTRAT,NOCRL1,NBVRL1,NTITE1,NCORL2,NBVRL2,    BOX10006
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,     BOX10007
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)           BOX10008
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, BOX10009
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10                             BOX10010
      COMMON/P3/NODNOT,UNOT,IBUBOF,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,   BOX10011
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,     BOX10012
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU, BOX10013
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,   BOX10014
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ       BOX10015
      COMMON/P4/SAVE,KBRAN,X1                                           BOX10016
      DIMENSION INT(ND1),ICC(ND1),IS(ND4),IBV(ND4),NBV(ND5),IUPPER(ND5) BOX10017
      DIMENSION BORIG(ND4),RHS(ND4),C2(ND1),BI(ND4),BN(ND5),U(ND6),     BOX10018
     1          PJ(ND4),SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),SLOLD(ND6),     BOX10019
     2          C2OLD(ND9),B(ND4,ND4)                                   BOX10020
      IF(ITRACE.GE.1)WRITE(6,1015)                                      BOX10021
      NODE=NODE + 1                                                     BOX10022
      IF(IOUTPT.NE.0)WRITE(6,1000)NODE,NBRNOD                           BOX10023
      IF(LNODE.EQ.2)GOTO290                                             BOX10024
C***********************************************************************BOX10025
C INITIALIZE THE DATA REQUIRED FOR SUBPROGRAM 1.                        BOX10026
C***********************************************************************BOX10027
      IF(XBRNOD.LT.SIGMAL(NBRVAR))XBRNOD=SIGMAL(NBRVAR)                 BOX10028
      IF(XBRNOD.GT.SIGMAU(NBRVAR))XBRNOD=SIGMAU(NBRVAR)                 BOX10029
C MODIFY THE UPPER LIMIT FOR THE BRANCHING VARIABLE.                    BOX10030
      SAVE=SIGMAU(NBRVAR)                                               BOX10031
      IF(INT(NBRVAR).NE.0)GOTO110                                       BOX10032
C BRANCHING ON A CONCAVE VARIABLE.                                      BOX10033
  100 IF(IOUTPT.NE.0)WRITE(6,1001)NBRVAR,ICC(NBRVAR)                    BOX10034
      SIGMAU(NBRVAR)=XBRNOD                                             BOX10035
      KBRAN=0                                                           BOX10036
      GOTO120                                                           BOX10037
C BRANCHING ON AN INTEGER VARIABLE.                                     BOX10038
  110 K=XBRNOD                                                          BOX10039
      X0=K                                                              BOX10040
      X1=K+1                                                            BOX10041
      IF(ABS(X0-XBRNOD).LE.TOL2 .AND. ICC(NBRVAR).NE.0)GOTO100          BOX10042
      IF(ABS(X1-XBRNOD).LE.TOL2 .AND. ICC(NBRVAR).NE.0)GOTO100          BOX10043
      IF(IOUTPT.NE.0)WRITE(6,1002)NBRVAR,INT(NBRVAR)                    BOX10044
      SIGMAU(NBRVAR)=X0                                                 BOX10045
      KBRAN=1                                                           BOX10046
C ESTABLISH THE UPPER LIMITS FOR SUBPROGRAM 1.                          BOX10047
  120 DO130J=1,N1P2                                                     BOX10048
  130 U(J)=SIGMAU(J) - SIGMAL(J)                                        BOX10049
C ESTABLISH THE COST DATA FOR SUBPROGRAM 1.                             BOX10050
      DO140J=1,N                                                        BOX10051
  140 TBRNOD=TBRNOD + C2(J)*(SIGMAL(J) - SLOLD(J))                      BOX10052
      TSIG=TBRNOD                                                       BOX10053
C ESTABLISH THE RIGHT-HAND-SIDE FOR SUBPROGRAM 1.                       BOX10054
      DO150I=1,M7                                                       BOX10055
  150 RHS(I)=BORIG(I)                                                   BOX10056
      DO180J=1,N1P2                                                     BOX10057
```

EXHIBIT 4 (Continued)

```
      IF(ABS(SIGMAL(J)).LE.EPSI)GOTO180                                 BOX10058
      DO160I=1,M7                                                       BOX10059
  160 PJ(I)=0.0                                                         BOX10060
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,  BOX10061
     1             J,NZEROS)                                            BOX10062
      DO170I1=1,NZEROS                                                  BOX10063
      I=IS(I1)                                                          BOX10064
  170 RHS(I)=RHS(I) - PJ(I)*SIGMAL(J)                                   BOX10065
      IF(IPHASE.EQ.2)RHS(MP2)=RHS(MP2) - PJ(MP2)*SIGMAL(J)              BOX10066
  180 CONTINUE                                                          BOX10067
C ESTABLISH THE BASIS INVERSE, THE VALUES OF THE BASIC VARIABLES, AND   BOX10068
C THE VALUES OF THE NONBASIC VARIABLES FOR SUBPROGRAM 1.                BOX10069
      DO190K=1,L10                                                      BOX10070
      IF(IUPPER(K).EQ.0)GOTO190                                         BOX10071
      J=NBV(K)                                                          BOX10072
      BN(K)=U(J)                                                        BOX10073
  190 CONTINUE                                                          BOX10074
      IF(LISTOP.EQ.1)GOTO270                                            BOX10075
      DO200I=1,M7                                                       BOX10076
  200 BI(I)=RHS(I)                                                      BOX10077
      DO230K=1,L10                                                      BOX10078
      IF(IUPPER(K).EQ.0)GOTO230                                         BOX10079
      J=NBV(K)                                                          BOX10080
      IF(ABS(U(J)).LE.EPSI)GOTO230                                      BOX10081
      DO210I=1,M7                                                       BOX10082
  210 PJ(I)=0.0                                                         BOX10083
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,  BOX10084
     1             J,NZEROS)                                            BOX10085
      DO220I1=1,NZEROS                                                  BOX10086
      I=IS(I1)                                                          BOX10087
  220 BI(I)=BI(I) - PJ(I)*U(J)                                          BOX10088
      BI(MP1)=BI(MP1) - PJ(MP1)*U(J)                                    BOX10089
      IF(IPHASE.EQ.2)BI(MP2)=BI(MP2) - PJ(MP2)*U(J)                     BOX10090
  230 CONTINUE                                                          BOX10091
      DO250I=1,M7                                                       BOX10092
      Q1=0.0                                                            BOX10093
      DO240J=1,M7                                                       BOX10094
  240 Q1=Q1 + B(I,J)*BI(J)                                              BOX10095
  250 PJ(I)=Q1                                                          BOX10096
      DO260I=1,M7                                                       BOX10097
  260 BI(I)=PJ(I)                                                       BOX10098
      GOTO280                                                           BOX10099
  270 CALL BINVRT (NZ,NF,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,  BOX10100
     1             BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)                  BOX10101
  280 REWIND 4                                                          BOX10102
      WRITE(4)(IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),    BOX10103
     1        (BI(I),I=1,ND4),(BN(I),I=1,ND5),L10,NITER,NBINV,          BOX10104
     2        ((B(I,J),I=1,ND4),J=1,ND4)                                BOX10105
C CHECK IF THE SUBPROGRAM IS INCLUDED IN THE NEXT SUBPROGRAM.           BOX10106
      IF(INT(NBRVAR).NE.0)GOTO370                                       BOX10107
      IF(ABS(U(NBRVAR)).GT.TOL2)GOTO370                                 BOX10108
      IF(ABS(SAVE - SIGMAL(NBRVAR)).LE.TOL2)GOTO370                     BOX10109
      IEOJ=1                                                            BOX10110
      IF(IOUTPT.NE.0)WRITE(6,1004)                                      BOX10111
      RETURN                                                            BOX10112
C***********************************************************************BOX10113
C INITIALIZE THE DATA REQUIRED FOR SUBPROGRAM 2.                        BOX10114
```

```
C***********************************************************************
  290 SIGMAU(NBRVAR)=SAVE
      SAVE=SIGMAL(NBRVAR)
      IF(KBRAN.EQ.1)GOTO300
      IF(IOUTPT.NE.0)WRITE(6,1001)NBRVAR,ICC(NBRVAR)
      SIGMAL(NBRVAR)=XBRNOD
      GOTO310
  300 IF(IOUTPT.NE.0)WRITE(6,1002)NBRVAR,INT(NBRVAR)
      SIGMAL(NBRVAR)=X1
  310 DELTA=SIGMAL(NBRVAR) - SAVE
C ADJUST THE UPPER LIMITS.
      U(NBRVAR)=SIGMAU(NBRVAR) - SIGMAL(NBRVAR)
C ADJUST THE COST DATA.
      IF(ITYPE.EQ.1)GOTO330
      DO320J=1,N
  320 C2(J)=C2OLD(J)
  330 TSIG=TBRNOD + C2(NBRVAR)*DELTA
C ADJUST THE RIGHT-HAND-SIDE.
      DO340I=1,M7
  340 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,NC2,ND3,ND4,ND5,
     1             NBRVAR,NZEROS)
      DO350I1=1,NZEROS
      I=IS(I1)
  350 RHS(I)=RHS(I) - PJ(I)*DELTA
      IF(IPHASE.EQ.2)RHS(MP2)=RHS(MP2) - PJ(MP2)*DELTA
C ADJUST THE VALUES OF THE BASIC VARIABLES.
      REWIND 4
      READ (4)(IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUFPER(I),I=1,ND5),
     1        (BI(I),I=1,ND4),(BN(I),I=1,ND5),L10,NITER,NBINV,
     2        ((B(I,J),I=1,ND4),J=1,ND4)
      DO360I=1,M7
      BI(I)=BI(I) + B(I,MP1)*PJ(MP1)*DELTA
      IF(NBRVAR.NE.IBV(I))GOTO360
      BI(I)=BI(I) - DELTA
  360 CONTINUE
C CHECK IF THE SUBPROGRAM IS INCLUDED IN THE LAST SUBPROGRAM.
      IF(INT(NBRVAR).NE.0)GOTO370
      IF(ABS(U(NBRVAR)).GT.TOL2)GOTO370
      IEOJ=1
      IF(IOUTPT.NE.0)WRITE(6,1005)
      RETURN
C***********************************************************************
C SOLVE SUBPROGRAM LNODE.
C***********************************************************************
  370 IEOJ=0
      DO390J=1,N1P2
      IF(U(J).LT.-TOL2)GOTO380
      IF(U(J).GE.TOL2*0.1)GOTO390
      U(J)=TOL2*0.1
      GOTO390
  380 IEOJ=1
  390 CONTINUE
      IF(IEOJ.EQ.0)GOTO410
C THE LOWER AND UPPER LIMITS ARE INCOMPATIBLE.
      IF(IOUTPT.NE.0)WRITE(6,1006)
      IF(IOUTPT.LE.1)RETURN
      WRITE(6,1007)                                                     BOX10172
      DO400J=1,N1P2                                                     BOX10173
      IF(U(J).GE.-TOL2)GOTO400                                          BOX10174
      WRITE(6,1008)J,SIGMAL(J),SIGMAU(J)                                BOX10175
  400 CONTINUE                                                          BOX10176
      RETURN                                                            BOX10177
C THE LOWER AND UPPER LIMITS ARE COMPATIBLE.                             BOX10178
  410 IF(ITYPE.EQ.2)GOTO460                                             BOX10179
C APPLY THE DUAL SIMPLEX ALGORITHM FIRST.                               BOX10180
      IF(IOUTPT.LE.1)GOTO450                                            BOX10181
      WRITE(6,1003)TSIG                                                 BOX10182
      WRITE(6,1009)                                                     BOX10183
      DO430J=1,N                                                        BOX10184
      K=INT(J)                                                          BOX10185
      L=ICC(J)                                                          BOX10186
      IF(IOUTPT.GE.3)GOTO420                                            BOX10187
      IF(K.EQ.0 .AND. L.EQ.0)GOTO430                                    BOX10188
  420 WRITE(6,1010)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)                      BOX10189
  430 CONTINUE                                                          BOX10190
      IF(IOUTPT.LE.2)GOTO450                                            BOX10191
      IF(NM1M3.EQ.N)GOTO450                                             BOX10192
      DO440J=NP1,NM1M3                                                  BOX10193
  440 WRITE(6,1011)J,SIGMAL(J),SIGMAU(J)                                BOX10194
  450 IALGO=2                                                           BOX10195
      CALL SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,BN, BOX10196
     1            U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)      BOX10197
      IF(IEOJ.NE.0)RETURN                                               BOX10198
  460 IF(ITYPE.EQ.1)GOTO580                                             BOX10199
C APPLY THE PRIMAL ALGORITHM SECOND.                                    BOX10200
C ESTABLISH NEW COST DATA.                                              BOX10201
      TSIG=0.0                                                          BOX10202
      DO490J=1,N                                                        BOX10203
      K=ICC(J)                                                          BOX10204
      IF(K.EQ.0)GOTO480                                                 BOX10205
      CALL GETOBJ (K,SIGMAL(J),F0)                                      BOX10206
      TSIG=TSIG + F0                                                    BOX10207
      IF(ABS(U(J)).LE.TOL2)GOTO470                                      BOX10208
      CALL GETOBJ (K,SIGMAU(J),F1)                                      BOX10209
      C2(J)=(F1 - F0)/U(J)                                              BOX10210
      GOTO490                                                           BOX10211
  470 C2(J)=0.0                                                         BOX10212
      GOTO490                                                           BOX10213
  480 TSIG=TSIG + C2(J)*SIGMAL(J)                                       BOX10214
  490 CONTINUE                                                          BOX10215
      IF(IOUTPT.LE.1)GOTO530                                            BOX10216
      WRITE(6,1003)TSIG                                                 BOX10217
      WRITE(6,1009)                                                     BOX10218
      DO510J=1,N                                                        BOX10219
      K=INT(J)                                                          BOX10220
      L=ICC(J)                                                          BOX10221
      IF(IOUTPT.GE.3)GOTO500                                            BOX10222
      IF(K.EQ.0 .AND. L.EQ.0)GOTO510                                    BOX10223
  500 WRITE(6,1010)J,K,L,SIGMAL(J),SIGMAU(J),C2(J)                      BOX10224
  510 CONTINUE                                                          BOX10225
      IF(IOUTPT.LE.2)GOTO530                                            BOX10226
      IF(NM1M3.EQ.N)GOTO530                                             BOX10227
      DO520J=NP1,NM1M3                                                  BOX10228
```

EXHIBIT 4 (Continued)

```
  520 WRITE(6,1011)J,SIGMAL(J),SIGMAU(J)                                BOX10229
  530 INDEX=0                                                           BOX10230
C BASIC VARIABLES.                                                      BOX10231
      DO550I=1,M7                                                       BOX10232
      IF(I.EQ.MP1)GOTO550                                               BOX10233
      J=IBV(I)                                                          BOX10234
      IF(J.GT.N)GOTO550                                                 BOX10235
      T=C2(J) - C2OLD(J)                                                BOX10236
      IF(ABS(T).LE.EPSI)GOTO550                                         BOX10237
      INDEX=1                                                           BOX10238
      BI(MP1)=BI(MP1) - T*BI(I)                                         BOX10239
      DO540J=1,M7                                                       BOX10240
  540 B(MP1,J)=B(MP1,J) - T*B(I,J)                                      BOX10241
  550 CONTINUE                                                          BOX10242
C NONBASIC VARIABLES.                                                   BOX10243
      DO560I=1,L10                                                      BOX10244
      J=NBV(I)                                                          BOX10245
      IF(J.GT.N)GOTO560                                                 BOX10246
      T=C2(J) - C2OLD(J)                                                BOX10247
      IF(ABS(T).LE.EPSI)GOTO560                                         BOX10248
      INDEX=1                                                           BOX10249
  560 CONTINUE                                                          BOX10250
      IF(INDEX.EQ.0)GOTO570                                             BOX10251
      IALGO=1                                                           BOX10252
      CALL SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,BN, BOX10253
     1        U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)          BOX10254
      IF(IEOJ.NE.0)RETURN                                               BOX10255
      GOTO580                                                           BOX10256
  570 IF(IOUTPT.GE.2)WRITE(6,1012)                                      BOX10257
  580 NFEAS=NFEAS + 1                                                   BOX10258
C PRINT THE SOLUTION.                                                   BOX10259
      Z=Z + TSIG                                                        BOX10260
      DO590J=1,N1P2                                                     BOX10261
  590 XZ(J)=XZ(J) + SIGMAL(J)                                           BOX10262
      IF(IOUTPT.EQ.0)GOTO600                                            BOX10263
      NUP=N                                                             BOX10264
      IF(IOUTPT.GE.3)NUP=NM1M3                                          BOX10265
      WRITE(6,1013)Z                                                    BOX10266
      WRITE(6,1014)(XZ(J),J=1,NUP)                                      BOX10267
  600 IF(NBVRUL.GE.3 .AND. NTIGHT.EQ.1)RETURN                           BOX10268
C DETERMINE THE SLOPES ASSOCIATED WITH THE OPTIMAL OBJECTIVE VALUE.     BOX10269
      CALL SLOPES (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,PJ,XJ,S0, BOX10270
     1             S1,B,ND1,ND2,ND3,ND4,ND5,ND8)                        BOX10271
      RETURN                                                            BOX10272
 1000 FORMAT(1H0,50(1H*)/6H0NODE ,I5/20H0BRANCHED FROM NODE ,I5)        BOX10273
 1001 FORMAT(23H0BRANCHING ON VARIABLE ,I5,27H WHICH IS CONCAVE VARIABLEBOX10274
     1 ,I5)                                                             BOX10275
 1002 FORMAT(23H0BRANCHING ON VARIABLE ,I5,27H WHICH IS INTEGER VARIABLEBOX10276
     1 ,I5)                                                             BOX10277
 1003 FORMAT(7H0TSIG =,E15.6)                                           BOX10278
 1004 FORMAT(76H0THE SUBPROGRAM NEED NOT BE SOLVED AS IT IS INCLUDED IN BOX10279
     1THE NEXT SUBPROGRAM.)                                             BOX10280
 1005 FORMAT(76H0THE SUBPROGRAM NEED NOT BE SOLVED AS IT IS INCLUDED IN BOX10281
     1THE LAST SUBPROGRAM.)                                             BOX10282
 1006 FORMAT(27H0THE PROGRAM IS INFEASIBLE.)                            BOX10283
 1007 FORMAT(38H0THE FOLLOWING LIMITS ARE INCOMPATIBLE/                 BOX10284
     1          9H0VARIABLE,9X,5HLOWER,12X,5HUPPER/                     BOX10285
```

EXHIBIT 4 (Continued)

```
     2         2X,6HNUMBER,10X,5HLIMIT,12X,5HLIMIT//)                      BOX10286
  1008 FORMAT(3X,I5,3X,E15.6,2X,E15.6)                                      BOX10287
  1009 FORMAT(9H0VARIABLE,4X,3HINT,8X,2HC0,12X,5HLOWER,12X,5HUPPER,13X,     BOX10288
     1         4HCOST/2X,6HNUMBER,3X,8HVARIABLE,2X,8HVARIABLE,9X,5HLIMIT,   BOX10289
     2         12X,5HLIMIT,9X,11HCOEFFICIENT/12X,6HNUMBER,4X,6HNUMBER//)    BOX10290
  1010 FORMAT(3X,I5,2(5X,I5),3X,3(E15.6,2X))                                BOX10291
  1011 FORMAT(3X,I5,23X,E15.6,2X,E15.6)                                     BOX10292
  1012 FORMAT(29H0THE LAST TABLEAU IS OPTIMAL.)                             BOX10293
  1013 FORMAT(17H0SOLUTION VALUE =,E15.6)                                   BOX10294
  1014 FORMAT(17H0VARIABLES        =,6E15.6/(17X,6E15.6))                   BOX10295
  1015 FORMAT(11H *****BOX15)                                               BOX10296
       END                                                                  BOX10297
```

```
      SUBROUTINE BOX17 (INT,ICC,IBV,C2,SIGMAL,SIGMAU,XZ,S0,S1,ND1,ND4,
     1                  ND6,ND8)
C DETERMINE LOWER BOUND. SELECT THE BRANCHING VARIABLE.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NFC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      COMMON/P5/IRCUNO
      DIMENSION INT(ND1),ICC(ND1),IBV(ND4)
      DIMENSION C2(ND1),SIGMAL(ND6),SIGMAU(ND6),XZ(ND6),S0(ND8),S1(ND8)
      IF(ITRACE.GE.1)WRITE(6,1009)
      IF(NBVRUL.EQ.5)GOTO420
      IF(NBVRUL.GE.3)GOTO250
C***********************************************************************
C MAXMIN AND MAXMAX BRANCHING VARIABLE SELECTION RULES.
C***********************************************************************
      BOUNDU=Z
      BOUNDL=Z
      PEN0=Z
      PEN1=Z
      PEN2=Z
      IBRV=0
      JBRV=0
      KBRV=0
      IFEAS=1
C BEGINNING OF LOOP.
      DO240I=1,M7
      IF(I.EQ.MP1)GOTO240
      J=IBV(I)
      IF(J.GT.N)GOTO240
      IF(ICC(J).NE.0)GOTO120
      IF(INT(J).EQ.0)GOTO240
C INTEGER LINEAR VARIABLE.
      K=XZ(J)
      X0=K
      X1=K+1
      IF(ABS(XZ(J)-X0).LE.TOL2)GOTO240
      IF(ABS(X1-XZ(J)).LE.TOL2)GOTO240
      IFEAS=0
      P0=BIGN
      P1=P0
      IF(S0(I).EQ.-BIGN)GOTO100
      P0=Z + S0(I)*(X0 - XZ(J))
  100 IF(S1(I).EQ.BIGN)GOTO110
      P1=Z + S1(I)*(X1 - XZ(J))
  110 IF(IBRV.EQ.0)IBRV=J
      IF(JBRV.NE.0)GOTO190
      JBRV=J
      JUPDN=2
      IF(P1.LE.P0)GOTO190
      JUPDN=1
```

EXHIBIT 4 (Continued)

```
      GOTO190
  120 IF(INT(J).NE.0)GOTO160
C CONCAVE VARIABLE.
  130 IF(ABS(XZ(J)-SIGMAL(J)).LE.TOL2)GOTO240
      IF(ABS(SIGMAU(J)-XZ(J)).LE.TOL2)GOTO240
      K=ICC(J)
      CALL GETOBJ (K,SIGMAL(J),F0)
      CALL GETOBJ (K,XZ(J),F1)
      DELTA=F1 - (F0 + C2(J)*(XZ(J)-SIGMAL(J)))
      BOUNDU=BOUNDU + DELTA
      P0=Z + DELTA
      P1=P0
      IF(S0(I).EQ.-BIGN)GOTO140
      T0=Z + S0(I)*(SIGMAL(J) - XZ(J))
      IF(T0.LT.P0)P0=T0
  140 IF(S1(I).EQ.BIGN)GOTO150
      T1=Z + S1(I)*(SIGMAU(J) - XZ(J))
      IF(T1.LT.P1)P1=T1
  150 IF(DELTA.EQ.0.0)GOTO190
      IF(IBRV.EQ.0)IBRV=J
      IF(JBRV.NE.0)GOTO190
      JBRV=J
      JUPDN=2
      IF(P1.LE.P0)GOTO190
      JUPDN=1
      GOTO190
C INTEGER CONCAVE VARIABLE.
  160 K=XZ(J)
      X0=K
      X1=K+1
      IF(ABS(XZ(J)-X0).LE.TOL2)GOTO130
      IF(ABS(X1-XZ(J)).LE.TOL2)GOTO130
      IFEAS=0
      K=ICC(J)
      CALL GETOBJ (K,SIGMAL(J),F0)
      CALL GETOBJ (K,X0,F1)
      CALL GETOBJ (K,X1,F2)
      P0=BIGN
      P1=P0
      IF(S0(I).EQ.-BIGN)GOTO170
      P0=Z + S0(I)*(X0-XZ(J)) + F1 - (F0 + C2(J)*(X0-SIGMAL(J)))
      IF(ABS(X0-SIGMAL(J)).LE.TOL2)GOTO170
      T0=Z + S0(I)*(SIGMAL(J)-XZ(J))
      IF(T0.LT.P0)P0=T0
  170 IF(S1(I).EQ.BIGN)GOTO180
      P1=Z + S1(I)*(X1-XZ(J)) + F2 - (F1 + C2(J)*(X1-SIGMAL(J)))
      IF(ABS(SIGMAU(J)-X1).LE.TOL2)GOTO180
      T1=Z + S1(I)*(SIGMAU(J)-XZ(J))
      IF(T1.LT.P1)P1=T1
  180 IF(IBRV.EQ.0)IBRV=J
      IF(JBRV.NE.0)GOTO190
      JBRV=J
      JUPDN=2
      IF(P1.LE.P0)GOTO190
      JUPDN=1
C ALL VARIABLE TYPES.
C DETERMINE THE LOWER BOUND.
```

EXHIBIT 4 (Continued)

```
  190 PENA=P0                                                           BOX10115
      IF(P1.LT.PENA)PENA=P1                                             BOX10116
      IF(PENA.LE.BOUNDL)GOTO200                                         BOX10117
      BOUNDL=PENA                                                       BOX10118
  200 IF((S0(I).EQ.-BIGN .OR. S1(I).EQ.BIGN) .AND. NTIGHT.EQ.0)GOTO240  BOX10119
C DETERMINE THE MAXMIN SELECTION.                                       BOX10120
      IF(PENA.LE.PEN0)GOTO210                                           BOX10121
      PEN0=PENA                                                         BOX10122
      IBRV=J                                                            BOX10123
C DETERMINE THE MAXMAX SELECTION.                                       BOX10124
  210 PENB=P0                                                           BOX10125
      JUD=2                                                             BOX10126
      IF(P1.LE.PENB)GOTO220                                             BOX10127
      PENB=P1                                                           BOX10128
      JUD=1                                                             BOX10129
  220 IF(PENB.LE.PEN1)GOTO230                                           BOX10130
      PEN1=PENB                                                         BOX10131
      JUPDN=JUD                                                         BOX10132
      JBRV=J                                                            BOX10133
C DETERMINE THE MAXMAX SELECTION TAKEN OVER THOSE VARIABLES FOR WHICH    BOX10134
C THE MIN IS GREATER THAN Z.                                            BOX10135
  230 IF(ABS(PENA - Z).LE.EPSI)GOTO240                                  BOX10136
      IF(PENB.LE.PEN2)GOTO240                                           BOX10137
      PEN2=PENB                                                         BOX10138
      KUPDN=JUD                                                         BOX10139
      KBRV=J                                                            BOX10140
  240 CONTINUE                                                          BOX10141
C END OF LOOP.                                                          BOX10142
      GOTO290                                                           BOX10143
C***********************************************************************BOX10144
C MIXED INTEGER LINEAR PROGRAM WITH THE MOST NONINTEGER                 BOX10145
C OR WEIGHTED NONINTEGER BRANCHING VARIABLE SELECTION RULE.             BOX10146
C***********************************************************************BOX10147
  250 BOUNDU=Z                                                          BOX10148
      BOUNDL=Z                                                          BOX10149
      PEN1=0.0                                                          BOX10150
      PEN2=0.0                                                          BOX10151
      JBRV=0                                                            BOX10152
      KBRV=0                                                            BOX10153
      IFEAS=1                                                           BOX10154
C BEGINNING OF LOOP.                                                    BOX10155
      DO280I=1,M7                                                       BOX10156
      IF(I.EQ.MP1)GOTO280                                               BOX10157
      J=IBV(I)                                                          BOX10158
      IF(J.GT.N)GOTO280                                                 BOX10159
      IF(INT(J).EQ.0)GOTO280                                            BOX10160
      K=XZ(J)                                                           BOX10161
      X0=K                                                              BOX10162
      P0=XZ(J) - X0                                                     BOX10163
      P1=1.0 - P0                                                       BOX10164
      IF(P0.LE.TOL2)GOTO280                                             BOX10165
      IF(P1.LE.TOL2)GOTO280                                             BOX10166
      IFEAS=0                                                           BOX10167
C DETERMINE THE MOST NONINTEGER SELECTION.                              BOX10168
      PENB=P0                                                           BOX10169
      JUD=1                                                             BOX10170
      IF(P1.GE.PENB)GOTO260                                             BOX10171
```

EXHIBIT 4 (Continued)

```
      PENB=P1
      JUD=2
  260 IF(PENB.LE.PEN1)GOTO270
      PEN1=PENB
      JUPDN=JUD
      JBRV=J
C DETERMINE THE WEIGHTED NONINTEGER SELECTION.
  270 PENB=PENB*ABS(C2(J))
      IF(PENB.LE.PEN2)GOTO280
      PEN2=PENB
      KUPDN=JUD
      KBRV=J
  280 CONTINUE
C END OF LOOP.
C*********************************************************************
C COMMON LOGIC FOR MAXMIN, MAXMAX, MOST NONINTEGER AND WEIGHTED
C NONINTEGER BRANCHING VARIABLE SELECTION RULES.
C*********************************************************************
  290 IF(ITYPE.NE.1)GOTO320
      IF(IROUND.EQ.0)GOTO320
C ROUND THE LOWER BOUND UP IF THE OBJECTIVE FUNCTION IS INTEGER VALUED.
      IF(BOUNDL.LE.0.0)GOTO300
      INTBD=BOUNDL + 1.0 + EPSIM
      GOTO310
  300 INTBD=BOUNDL + EPSIM
  310 BOUNDL=INTBD
  320 IF(KBRV.NE.0)GOTO330
      KBRV=JBRV
      KUPDN=JUPDN
C SELECT THE BRANCHING VARIABLE FOR A ONE PHASE METHOD OR PHASE 1 OF
C A TWO PHASE METHOD.
  330 GOTO(340,350,360,350,350),NBVRL1 + 1
  340 IBRVR1=IBRV
      IUPDN1=2
      GOTO370
  350 IBRVR1=JBRV
      IUPDN1=JUPDN
      GOTO370
  360 IBRVR1=KBRV
      IUPDN1=KUPDN
  370 XBRVR1=0.0
      IF(IBRVR1.NE.0)XBRVR1=XZ(IBRVR1)
      IF(NSTRAT.EQ.1)GOTO440
C SELECT THE BRANCHING VARIABLE FOR PHASE 2 OF A TWO PHASE METHOD.
      GOTO(380,390,400,390,400),NBVRL2 + 1
  380 IBRVR2=IBRV
      IUPDN2=2
      GOTO410
  390 IBRVR2=JBRV
      IUPDN2=JUPDN
      GOTO410
  400 IBRVR2=KBRV
      IUPDN2=KUPDN
  410 XBRVR2=0.0
      IF(IBRVR2.NE.0)XBRVR2=XZ(IBRVR2)
      GOTO440
C*********************************************************************
```

AD-A062 012 NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA F/G 12/2
SOLUTION OF THE INTEGER CONCAVE PROGRAM USING THE IC PHI N ALGO--ETC(U)
NOV 78 H W LOOMIS
UNCLASSIFIED NSWC/DL-TR-3120-VOL-2 NL

2 OF 2
AD
A062012

END
DATE
FILMED
3-79
DDC

EXHIBIT 4 (Continued)

```
C CONCAVE NONLINEAR PROGRAM WITH THE CONVENTIONAL                     BOX10229
C BRANCHING VARIABLE SELECTION RULE.                                  BOX10230
C*********************************************************************BOX10231
  420 BOUNDU=Z                                                        BOX10232
      PEN=0.0                                                         BOX10233
      IBRVR1=0                                                        BOX10234
      DO430I=1,M7                                                     BOX10235
      IF(I.EQ.MP1)GOTO430                                             BOX10236
      J=IBV(I)                                                        BOX10237
      IF(J.GT.N)GOTO430                                               BOX10238
      K=ICC(J)                                                        BOX10239
      IF(K.EQ.0)GOTO430                                               BOX10240
      CALL GETOBJ (K,SIGMAL(J),F0)                                    BOX10241
      CALL GETOBJ (K,XZ(J),F1)                                        BOX10242
      DELTA=F1 - (F0 + C2(J)*(XZ(J)-SIGMAL(J)))                       BOX10243
      BOUNDU=BOUNDU + DELTA                                           BOX10244
      IF(DELTA.LE.PEN)GOTO430                                         BOX10245
      PEN=DELTA                                                       BOX10246
      IBRVR1=J                                                        BOX10247
  430 CONTINUE                                                        BOX10248
      XBRVR1=0.0                                                      BOX10249
      IF(IBRVR1.NE.0)XBRVR1=XZ(IBRVR1)                                BOX10250
      BOUNDL=Z                                                        BOX10251
      IFEAS=1                                                         BOX10252
C*********************************************************************BOX10253
C PRINT OUT THE RESULTS.                                              BOX10254
C*********************************************************************BOX10255
  440 IF(IOUTPT.EQ.0)GOTO540                                          BOX10256
      IF(NSTRAT.EQ.2 .AND. LPHASE.EQ.2)GOTO500                        BOX10257
      IF(IBRVR1.EQ.0)GOTO530                                          BOX10258
      IF(1.LE.NBVRL1 .AND. NBVRL1.LE.4)GOTO450                        BOX10259
      WRITE(6,1000)IBRVR1                                             BOX10260
      GOTO470                                                         BOX10261
  450 IF(IUPDN1.EQ.2)GOTO460                                          BOX10262
      WRITE(6,1001)IBRVR1                                             BOX10263
      GOTO470                                                         BOX10264
  460 WRITE(6,1002)IBRVR1                                             BOX10265
  470 IF(NSTRAT.EQ.1)GOTO530                                          BOX10266
      IF(IBRVR2.EQ.0)GOTO530                                          BOX10267
      IF(1.LE.NBVRL2 .AND. NBVRL2.LE.4)GOTO480                        BOX10268
      WRITE(6,1003)IBRVR2                                             BOX10269
      GOTO530                                                         BOX10270
  480 IF(IUPDN2.EQ.2)GOTO490                                          BOX10271
      WRITE(6,1004)IBRVR2                                             BOX10272
      GOTO530                                                         BOX10273
  490 WRITE(6,1005)IBRVR2                                             BOX10274
      GOTO530                                                         BOX10275
  500 IF(IBRVR2.EQ.0)GOTO530                                          BOX10276
      IF(1.LE.NBVRL2 .AND. NBVRL2.LE.4)GOTO510                        BOX10277
      WRITE(6,1000)IBRVR2                                             BOX10278
      GOTO530                                                         BOX10279
  510 IF(IUPDN2.EQ.2)GOTO520                                          BOX10280
      WRITE(6,1001)IBRVR2                                             BOX10281
      GOTO530                                                         BOX10282
  520 WRITE(6,1002)IBRVR2                                             BOX10283
  530 WRITE(6,1006)BOUNDL                                             BOX10284
      IF(IFEAS.EQ.1)WRITE(6,1007)BOUNDU                               BOX10285
```

EXHIBIT 4 (Continued)

```
  540 IF(NODE.NE.1 .OR. IBUBOP.EQ.0)RETURN                                BOX10286
C ESTABLISH THE INITIAL BEST UPPER BOUND.                                 BOX10287
      UNOT=BOUNDL + PCBUB                                                 BOX10288
      IF(IOUTPT.NE.0)WRITE(6,1008)UNOT                                    BOX10289
      RETURN                                                              BOX10290
 1000 FORMAT(10H0VARIABLE ,I5,27H IS THE BRANCHING VARIABLE.)             BOX10291
 1001 FORMAT(10H0VARIABLE ,I5,69H IS THE BRANCHING VARIABLE. CONTINUE BRBOX10292
     1ANCHING FROM THE LOWER BRANCH.)                                     BOX10293
 1002 FORMAT(10H0VARIABLE ,I5,69H IS THE BRANCHING VARIABLE. CONTINUE BRBOX10294
     1ANCHING FROM THE UPPER BRANCH.)                                     BOX10295
 1003 FORMAT(10H0VARIABLE ,I5,39H IS THE BRANCHING VARIABLE FOR PHASE 2.BOX10296
     1)                                                                   BOX10297
 1004 FORMAT(10H0VARIABLE ,I5,81H IS THE BRANCHING VARIABLE FOR PHASE 2.BOX10298
     1 CONTINUE BRANCHING FROM THE LOWER BRANCH.)                         BOX10299
 1005 FORMAT(10H0VARIABLE ,I5,81H IS THE BRANCHING VARIABLE FOR PHASE 2.BOX10300
     1 CONTINUE BRANCHING FROM THE UPPER BRANCH.)                         BOX10301
 1006 FORMAT(14H0LOWER BOUND =,E15.6)                                     BOX10302
 1007 FORMAT(14H0UPPER BOUND =,E15.6)                                     BOX10303
 1008 FORMAT(31H0THE PHASE 1 BEST UPPER BOUND =,E15.6)                    BOX10304
 1009 FORMAT(11H *****BOX17)                                              BOX10305
      END                                                                 BOX10306
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX23 (INUSE,XNOT,CAPP,XZ,ND1,ND6,ND10)
C UPDATE THE BEST UPPER BOUND. IF IN PHASE 1, MERGE THE SUBLIST INTO
C THE LIST AND ENTER PHASE 2. EDIT THE LIST.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(1 )
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION INUSE(ND10)
      DIMENSION XNOT(ND1),CAPP(ND10),XZ(ND6)
      IF(ITRACE.GE.1)WRITE(6,1004)
C UPDATE THE BEST UPPER BOUND.
      NODNOT=NODE
      UNOT=BOUNDU
      DO100J=1,N
  100 XNOT(J)=XZ(J)
C EDIT THE LIST.
      NDELET=0
      DO110I=1,MXLIST
      IF(INUSE(I).EQ.0)GOTO110
      IF(CAPP(I).LT.(1.-TOL1)*UNOT)GOTO110
      INUSE(I)=0
      NDELET=NDELET+1
  110 CONTINUE
      IF(IOUTPT.NE.0)WRITE(6,1000)NODNOT,UNOT
      IF(NDELET.EQ.0)GOTO120
      NLIST=NLIST-NDELET
      IF(IOUTPT.EQ.0)GOTO120
      WRITE(6,1001)NDELET
      WRITE(6,1002)NLIST
  120 IF(ITYPE.EQ.2)RETURN
      IF(LPHASE.EQ.2)RETURN
C ENTER PHASE 2.
      LPHASE=2
      IF(NSTRAT.EQ.1)GOTO130
      NODRUL=NODRL2
      NBVRUL=NBVRL2
      NTIGHT=NTITE2
  130 IF(IOUTPT.NE.0)WRITE(6,1003)
      IF(IBUBOP.EQ.0)RETURN
      IBUBOP=0
C MERGE THE SUBLIST INTO THE LIST.
      DO140I=1,MXLIST
      IF(INUSE(I).GE.0)GOTO140
      INUSE(I)=-INUSE(I)
  140 CONTINUE
      RETURN
 1000 FORMAT(6H0NODE ,I5,35H PROVIDES THE NEW BEST UPPER BOUND ,E15.6)
 1001 FORMAT(46H0THE NUMBER OF NODES DELETED FROM THE LIST IS ,I5)
 1002 FORMAT(26H0THE CURRENT LIST SIZE IS ,I5)
 1003 FORMAT(14H0ENTER PHASE 2)
 1004 FORMAT(11H *****BOX23)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BOX25 (INUSE,IMS,C2,CAPP,CAPL,SIGMAL,SIGMAU,SLOLD,
     1                  SUOLD,C2OLD,FMS,ND1,ND6,ND9,ND10,ND11,NDMS2,
     2                  NDMS3)
C STORE NODE LNODE IN THE SUBLIST (WHEN IN PHASE 1 AND LOWER BOUND LESS
C THAN BEST UPPER BOUND) OR IN THE LIST (OTHERWISE).
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M1.
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,ELB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IZOJ
      DIMENSION INUSE(ND10),IMS(NDMS2)
      DIMENSION C2(ND1),CAPP(ND10),CAPL(ND11),SIGMAL(ND6),SIGMAU(ND6),
     1          SLOLD(ND6),SUOLD(ND6),C2OLD(ND9),FMS(NDMS3)
      IF(ITRACE.GE.1)WRITE(6,1009)
C CHECK IF THE MAXIMUM LIST SIZE WILL BE EXCEEDED.
      IF(NLIST+1.LE.MXLIST)GOTO130
      IF(IOUTPT.NE.0)WRITE(6,1000)NODE
C DETERMINE THAT NODE IN THE LIST WITH THE GREATEST LOWER BOUND.
      GLB=-BIGN
      DO100I=1,MXLIST
      IF(INUSE(I).EQ.0)GOTO100
      IF(CAPP(I).LE.GLB)GOTO100
      GLB=CAPP(I)
      I0=I
  100 CONTINUE
      IF(BOUNDL.LT.GLB)GOTO110
      IF(IOUTPT.EQ.0)RETURN
      WRITE(6,1001)
      RETURN
  110 IF(IOUTPT.NE.0)WRITE(6,1002)INUSE(I0)
      IF(INUSE(I0).GE.0)GOTO120
      NLISTS=NLISTS - 1
  120 NLIST=NLIST - 1
      INUSE(I0)=0
      GOTO150
C FIND AVAILABLE SPACE IN THE LIST.
  130 DO140I0=1,MXLIST
      IF(INUSE(I0).EQ.0)GOTO150
  140 CONTINUE
  150 IF(IBUBOP.EQ.0 .OR. BOUNDL.GE.UNOT)GOTO160
C***********************************************************************
C STORE THE NODE IN THE SUBLIST.
C***********************************************************************
      INUSE(I0)=-NODE
      NLISTS=NLISTS + 1
      GOTO170
C***********************************************************************
C STORE THE NODE IN THE LIST.
C***********************************************************************
  160 INUSE(I0)=NODE
  170 NLIST=NLIST + 1
      IF(NLIST.GT.LSTMX)LSTMX=NLIST
```

EXHIBIT 4 (Continued)

```
      IF(IOUTPT.EQ.0)GOTO180                                            BOX20058
      WRITE(6,1003)NODE                                                 BOX20059
      WRITE(6,1004)NLIST                                                BOX20060
C***********************************************************************BOX20061
C SET CAPP AND CAPL.                                                    BOX20062
C***********************************************************************BOX20063
C SET CAPP FOR THE NODE BEING SAVED.                                    BOX20064
  180 CAPP(IO)=BOUNDL                                                   BOX20065
      IF(NSTRAT.EQ.2)GOTO190                                            BOX20066
      IF(NODRL1.EQ.1)GOTO200                                            BOX20067
      GOTO280                                                           BOX20068
  190 IF(NODRL2.EQ.1)GOTO200                                            BOX20069
      IF(LPHASE.EQ.1 .AND. NODRL1.EQ.1)GOTO200                          BOX20070
      GOTO280                                                           BOX20071
C DETERMINE CAPL FOR THE NODE BEING SAVED.                              BOX20072
  200 IF(1.LE.NBVRUL .AND. NBVRUL.LE.4)GOTO220                          BOX20073
      IF(LNODE.EQ.2)GOTO210                                             BOX20074
      PNEWND=PBRNOD                                                     BOX20075
      GOTO270                                                           BOX20076
  210 PNEWND=PBRNOD + 1.0                                               BOX20077
      GOTO240                                                           BOX20078
  220 IF(LNODE.EQ.NUPDWN)GOTO230                                        BOX20079
      PNEWND=PBRNOD                                                     BOX20080
      GOTO270                                                           BOX20081
  230 PNEWND=PBRNOD + 1.0                                               BOX20082
  240 IF(IBUBOP.EQ.0)GOTO270                                            BOX20083
      PMIN=BIGN                                                         BOX20084
      DO250I=1,MXLIST                                                   BOX20085
      IF(I.EQ.IO)GOTO250                                                BOX20086
      IF(INUSE(I).EQ.0)GOTO250                                          BOX20087
      IF(CAPL(I).LE.PBRNOD)GOTO250                                      BOX20088
      IF(CAPL(I).GE.PMIN)GOTO250                                        BOX20089
      PMIN=CAPL(I)                                                      BOX20090
  250 CONTINUE                                                          BOX20091
      IF(PMIN.GT.PNEWND)GOTO270                                         BOX20092
C INCREMENT CAPL FOR NODES SUBSEQUENT TO THE NODE BEING SAVED.          BOX20093
      DO260I=1,MXLIST                                                   BOX20094
      IF(INUSE(I).EQ.0)GOTO260                                          BOX20095
      IF(CAPL(I).LE.PBRNOD)GOTO260                                      BOX20096
      CAPL(I)=CAPL(I)+1.0                                               BOX20097
  260 CONTINUE                                                          BOX20098
C SET CAPL FOR THE NODE BEING SAVED.                                    BOX20099
  270 CAPL(IO)=PNEWND                                                   BOX20100
C***********************************************************************BOX20101
C WRITE OUT THE DATA FOR THIS NODE.                                     BOX20102
C***********************************************************************BOX20103
  280 IMS(1)=IBRVR1                                                     BOX20104
      IMS(2)=IUPDN1                                                     BOX20105
      IMS(3)=L10                                                        BOX20106
      IMS(4)=NITER                                                      BOX20107
      IMS(5)=NBINV                                                      BOX20108
      IMS(6)=M7                                                         BOX20109
      IMS(7)=IPHASE                                                     BOX20110
      IMS(8)=NPHASE                                                     BOX20111
      IMS(9)=NM3M7                                                      BOX20112
      IF(NSTRAT.EQ.1)GOTO290                                            BOX20113
      IMS(10)=IBRVR2                                                    BOX20114
```

EXHIBIT 4 (Continued)

```
      IMS(11)=IUPON2                                                    BOX20115
  290 FMS(1)=Z                                                          BOX20116
      FMS(2)=TSIG                                                       BOX20117
      FMS(3)=XBRVR1                                                     BOX20118
      IF(NSTRAT.EQ.1)GOTO300                                            BOX20119
      FMS(4)=XBRVR2                                                     BOX20120
  300 IF(LNODE.EQ.2)GOTO330                                             BOX20121
C INTERCHANGE SIGMAL,SLOLD AND SIGMAU,SUOLD.                             BOX20122
      DO310J=1,N1P2                                                     BOX20123
      T1=SLOLD(J)                                                       BOX20124
      T2=SUOLD(J)                                                       BOX20125
      SLOLD(J)=SIGMAL(J)                                                BOX20126
      SUOLD(J)=SIGMAU(J)                                                BOX20127
      SIGMAL(J)=T1                                                      BOX20128
  310 SIGMAU(J)=T2                                                      BOX20129
      IF(ITYPE.EQ.1)GOTO360                                             BOX20130
C INTERCHANGE C2 AND C2OLD.                                             BOX20131
      DO320J=1,N                                                        BOX20132
      TEMP=C2OLD(J)                                                     BOX20133
      C2OLD(J)=C2(J)                                                    BOX20134
  320 C2(J)=TEMP                                                        BOX20135
      GOTO360                                                           BOX20136
C PUT SIGMAL INTO SLOLD, SIGMAU INTO SUOLD.                             BOX20137
  330 DO340J=1,N1P2                                                     BOX20138
      SLOLD(J)=SIGMAL(J)                                                BOX20139
  340 SUOLD(J)=SIGMAU(J)                                                BOX20140
      IF(ITYPE.EQ.1)GOTO360                                             BOX20141
C PUT C2 INTO C2OLD.                                                    BOX20142
      DO350J=1,N                                                        BOX20143
  350 C2OLD(J)=C2(J)                                                    BOX20144
  360 CALL WRITMS (2,IMS,NDMS2,IG)                                      BOX20145
      CALL WRITMS (3,FMS,NDMS3,IG)                                      BOX20146
      IF(LNODE.EQ.2)GOTO390                                             BOX20147
C INTERCHANGE SIGMAL,SLOLD AND SIGMAU,SUOLD.                             BOX20148
      DO370J=1,N1P2                                                     BOX20149
      T1=SLOLD(J)                                                       BOX20150
      T2=SUOLD(J)                                                       BOX20151
      SLOLD(J)=SIGMAL(J)                                                BOX20152
      SUOLD(J)=SIGMAU(J)                                                BOX20153
      SIGMAL(J)=T1                                                      BOX20154
  370 SIGMAU(J)=T2                                                      BOX20155
      IF(ITYPE.EQ.1)GOTO390                                             BOX20156
C INTERCHANGE C2 AND C2OLD.                                             BOX20157
      DO380J=1,N                                                        BOX20158
      TEMP=C2OLD(J)                                                     BOX20159
      C2OLD(J)=C2(J)                                                    BOX20160
  380 C2(J)=TEMP                                                        BOX20161
C***********************************************************************BOX20162
C PRINT OUT THE LIST.                                                   BOX20163
C***********************************************************************BOX20164
  390 IF(IOUTPT.LE.2)RETURN                                             BOX20165
      INDEX=0                                                           BOX20166
      IF(NSTRAT.EQ.2)GOTO400                                            BOX20167
      IF(NODRL1.EQ.1)GOTO420                                            BOX20168
      GOTO410                                                           BOX20169
  400 IF(NODRL2.EQ.1)GOTO420                                            BOX20170
      IF(LPHASE.EQ.1 .AND. NODRL1.EQ.1)GOTO420                          BOX20171
```

EXHIBIT 4 (Continued)

```
 410 INDEX=1                                                               BOX20172
     WRITE(6,1005)                                                         BOX20173
     GOTO430                                                               BOX20174
 420 WRITE(6,1006)                                                         BOX20175
 430 DO045.I=1,MXLIST                                                      BOX20176
     IF(INUSE(I).EQ.0)GOTO450                                              BOX20177
     IF(INDEX.EQ.0)GOTO440                                                 BOX20178
     WRITE(6,1007)INUSE(I),CAPP(I)                                         BOX20179
     GOTO450                                                               BOX20180
 440 WRITE(6,1008)INUSE(I),CAPP(I),CAPL(I)                                 BOX20181
 450 CONTINUE                                                              BOX20182
     RETURN                                                                BOX20183
1000 FORMAT(48HOLIST SIZE EXCEEDED WITH ATTEMPT TO RECORD NODE ,I5)        BOX20184
1001 FORMAT(29H THE NODE IS NOT BEING SAVED.)                              BOX20185
1002 FORMAT(6H NODE  ,I5,31H IS BEING PURGED FROM THE LIST.)               BOX20186
1003 FORMAT(6H NODE  ,I5,19H SAVED IN THE LIST.)                           BOX20187
1004 FORMAT(26HOTHE CURRENT LIST SIZE IS ,I5)                              BOX20188
1005 FORMAT(1H0,3X,4HNODE,9X,5HLOWER/17X,5HBOUND//)                        BOX20189
1006 FORMAT(1H0,3X,4HNODE,9X,5HLOWER,8X,10HPROCESSING/17X,5HBOUND,         BOX20190
   1        11X,5HORDER//)                                                 BOX20191
1007 FORMAT(3X,I5,2X,E15.6)                                                BOX20192
1008 FORMAT(3X,I5,2X,E15.6,2X,F11.0)                                       BOX20193
1009 FORMAT(11H *****BCX25)                                                BOX20194
     END                                                                   BOX20195
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE ADJUST (V1,V2,V3,V4,T1,T2,T3,T4,SL,SU)
C ADJUST THE LOWER AND UPPER LIMITS ON A VARIABLE USING THE BEST
C UPPER BOUND.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MEINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,YBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      IF(ITRACE.GE.2)WRITE(6,1000)
C ASSUME THAT THE CURVE CONSISTS OF TWO LINEAR SEGMENTS, ONE CONNECTING
C THE POINTS (V1,T1) AND (V2,T2), THE OTHER CONNECTING THE POINTS
C (V3,T3) AND (V4,T4).
      SL=V1
      IF(T1.LE.UNOT)GOTO130
      IF(T2.GE.UNOT)GOTO100
      SL=V2 + (UNOT - T2)*(V1 - V2)/(T1 - T2)
      GOTO130
  100 IF(T3.GT.UNOT)GOTO110
      SL=V3
      GOTO130
  110 IF(T4.GE.UNOT)GOTO120
      SL=V4 + (UNOT - T4)*(V3 - V4)/(T3 - T4)
      GOTO130
  120 SL=V4
  130 SU=V4
      IF(T4.LE.UNOT)RETURN
      IF(T3.GE.UNOT)GOTO140
      SU=V3 + (UNOT - T3)*(V3 - V4)/(T3 - T4)
      RETURN
  140 IF(T2.GT.UNOT)GOTO150
      SU=V2
      RETURN
  150 IF(T1.GE.UNOT)GOTO160
      SU=V1 + (UNOT - T1)*(V1 - V2)/(T1 - T2)
      RETURN
  160 SU=V1
      RETURN
 1000 FORMAT(12H *****ADJUST)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE BINVRT (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,
     1                   U,PJ,BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)
C COMPUTE THE BASIS INVERSE CORRESPONDING TO THE BASIS SPECIFIED
C IN ARRAY IBV.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLE,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XERNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION IS(ND4),IBV(ND4),NBV(ND5),IUPPER(ND5)
      DIMENSION BI(ND4),U(ND6),PJ(ND4),B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1000)
C INITIALIZE THE BASIS MATRIX.
      DO100I=1,M7
      DO100J=1,M7
  100 B(I,J)=0.0
      DO130J=1,M7
      KIND=IBV(J)
      DO110I=1,M7
  110 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             KIND,NZEROS)
      DO120I1=1,NZEROS
      I=IS(I1)
  120 B(I,J)=PJ(I)
      B(MP1,J)=PJ(MP1)
      IF(IPHASE.EQ.2)B(MP2,J)=PJ(MP2)
  130 CONTINUE
C INITIALIZE THE RIGHT-HAND-SIDE, ADJUSTING FOR VARIABLES AT UPPER
C BOUND IF NECESSARY.
      NRH=N+M3+M7+1
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             NRH,NZEROS)
      DO140J=1,M7
  140 BI(J)=PJ(J)
      DO170K=1,L10
      IF(IUPPER(K).EQ.0)GOTO170
      INDEX=NBV(K)
      DO150I=1,M7
  150 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             INDEX,NZEROS)
      DO160I1=1,NZEROS
      I=IS(I1)
  160 BI(I)=BI(I) - PJ(I)*U(INDEX)
      BI(MP1)=BI(MP1) - PJ(MP1)*U(INDEX)
      IF(IPHASE.EQ.2)BI(MP2)=BI(MP2) - PJ(MP2)*U(INDEX)
  170 CONTINUE
C OBTAIN THE BASIS INVERSE AND THE CORRESPONDING RIGHT-HAND-SIDE.
      CALL INVERT (IS,BI,PJ,BINV,B,ND4,ND7)
      RETURN
 1000 FORMAT(12H *****BINVRT)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,
     1                   ND4,ND5,J,NZEROS)
C GET THE J-TH COLUMN FROM THE CONSTRAINT MATRIX.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,MFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PERNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      DIMENSION NZ(ND1),NP(ND1),IR(ND2),IA(ND2),IS(ND4)
      DIMENSION TC(ND3),RHS(ND4),C2(ND1),C1(ND5),PJ(ND4)
      IF(ITRACE.GE.2)WRITE(6,1000)
      IF(J.GT.N)GOTO110
      NZEROS=NZ(J)
      NPOINT=NP(J)
      DO100K=1,NZEROS
      NPOINT=NPOINT+1
      I=IR(NPOINT)
      IS(K)=I
      INDEX=IA(NPOINT)
  100 PJ(I)=TC(INDEX)
      PJ(MP1)=C2(J)
      IF(IPHASE.EQ.2)PJ(MP2)=C1(J)
      RETURN
  110 IF(J.GT.NM3)GOTO120
      J1=J-NM1M2
      NZEROS=1
      IS(1)=J1
      PJ(J1)=-1.0
      IF(IPHASE.EQ.2)PJ(MP2)=C1(J)
      RETURN
  120 IF(J.GT.NM3M7)GOTO130
      J1=J-NM3
      NZEROS=1
      IS(1)=J1
      PJ(J1)=1.0
      RETURN
  130 DO140I=1,M
  140 PJ(I)=RHS(I)
      PJ(MP1)=RHS(MP1)
      IF(IPHASE.EQ.2)PJ(MP2)=RHS(MP2)
      RETURN
 1000 FORMAT(12H *****GETCOL)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE INPUT1 (NZ,NF,ND1)
C READ THE NUMBER OF LESS THAN, EQUALITY, GREATER THAN CONSTRAINTS.
C READ THE NUMBER OF NONZERO ENTRIES BY COLUMN.  DEVELOP THE COLUMN
C POINTERS AND TOTAL STORAGE REQUIRED.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      DIMENSION NZ(ND1),NP(ND1)
      IF(ITRACE.GE.1)WRITE(6,1004)
      READ(ITAPE,1000)M1,M2,M3
      WRITE(6,1001)M1,M2,M3
      M4=M2+M3
      N1=N+(M3+M1)+(M2+M3)
      MP1=M+1
      MP2=M+2
      NM3=N+M3
      NM1M2=N-M1-M2
      NM1M3=N+M1+M3
      N1P2=N1+2
      NP1=N+1
      READ(ITAPE,1000)(NZ(J),J=1,N)
      WRITE(6,1002)(NZ(J),J=1,N)
      NP(1)=0
      DO100J=2,N
  100 NP(J)=NP(J-1)+NZ(J-1)
      NSUM=NP(N)+NZ(N)
      PERCT=NSUM
      DENOM=M*N
      PERCT=100.*PERCT/DENOM
      WRITE(6,1003)NSUM,PERCT
      RETURN
 1000 FORMAT(16I5)
 1001 FORMAT(38H0NUMBER OF CONSTRAINTS BY TYPE          =,3I5)
 1002 FORMAT(38H0NUMBER OF NONZERO ENTRIES BY COLUMN =,16I5/(38X,16I5))
 1003 FORMAT(38H0TOTAL NUMBER OF NONZERO ENTRIES        =,I10,
     1      15H (A DENSITY OF ,F5.1,10H PERCENT).)
 1004 FORMAT(12H *****INPUT1)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE INPUT2 (NZ,NP,IR,IA,ND1,ND2)
C READ THE CONSTRAINT MATRIX COLUMN-BY-COLUMN.  IT IS ASSUMED THAT
C THE CONSTRAINTS ARE ORDERED (LESS THAN, EQUALITY, GREATER THAN
C CONSTRAINTS).
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      DIMENSION NZ(ND1),NP(ND1),IR(ND2),IA(ND2)
      IF(ITRACE.GE.1)WRITE(6,1003)
      DO100J=1,N
      K1=NP(J)+1
      K2=NP(J)+NZ(J)
      READ(ITAPE,1000)(IR(K),IA(K),K=K1,K2)
  100 WRITE(6,1001)J,(IR(K),IA(K),K=K1,K2)
      READ(ITAPE,1000)NTC
      WRITE(6,1002)NTC
      RETURN
 1000 FORMAT(16I5)
 1001 FORMAT(8H0COLUMN ,I5,16H, ROW/CONSTANT =,16I5/(29X,16I5))
 1002 FORMAT(22H0NUMBER OF CONSTANTS =,I5)
 1003 FORMAT(12H *****INPUT2)
      END
      SUBROUTINE INPUT3 (INT,ICC,NV,TC,BORIG,C2,SIGMAL,SIGMAU,V,ND1,ND3,
     1                   ND4,ND6)
C READ THE TABLE OF CONSTANTS, THE RIGHT-HAND-SIDE, THE LOWER AND
C UPPER BOUNDS, THE COST DATA, AND THE LISTS OF INTEGER AND CONCAVE
C VARIABLES.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P5/IROUND
      DIMENSION INT(ND1),ICC(ND1),NV(ND6)
      DIMENSION TC(ND3),BORIG(ND4),C2(ND1),SIGMAL(ND6),SIGMAU(ND6),
     1          V(ND6)
      DATA BIGINT/1.0E+14/
      IF(ITRACE.GE.1)WRITE(6,1014)
C READ THE TABLE OF CONSTANTS AND THE RIGHT-HAND-SIDES.  IT IS ASSUMED
C THAT THE RIGHT-HAND-SIDES ARE NONNEGATIVE.
      READ(ITAPE,1001)(TC(K),K=1,NTC)
      READ(ITAPE,1001)(BORIG(I),I=1,M)
      IF(ITAPE.NE.5)REWIND ITAPE
      BORIG(MP1)=0.0
      WRITE(6,1002)(TC(K),K=1,NTC)
      WRITE(5,1003)(BORIG(I),I=1,M)
C READ LOWER AND UPPER BOUNDS ON THE VARIABLES.
      DO100J=1,N1P2
      SIGMAL(J)=0.0
  100 SIGMAU(J)=BIGN
      READ(5,1000)NDN
      WRITE(6,1004)NDN
      IF(NDN.EQ.0)GOTO120
      READ(5,1000)(NV(K),K=1,NDN)
      READ(5,1001)(V(K),K=1,NDN)
      WRITE(5,1005)(NV(K),K=1,NDN)
      WRITE(5,1006)(V(K),K=1,NDN)
      DO110K=1,NDN
      J=NV(K)
  110 SIGMAL(J)=V(K)
  120 READ(5,1000)NUP
      WRITE(5,1007)NUP
      IF(NUP.EQ.0)GOTO140
      READ(5,1000)(NV(K),K=1,NUP)
      READ(5,1001)(V(K),K=1,NUP)
      WRITE(5,1005)(NV(K),K=1,NUP)
      WRITE(5,1008)(V(K),K=1,NUP)
      DO130K=1,NUP
      J=NV(K)
  130 SIGMAU(J)=V(K)
C READ COST DATA.
  140 READ(5,1001)(C2(J),J=1,N)
      WRITE(5,1009)(C2(J),J=1,N)
      DO150J=1,N
  150 INT(J)=0
      IF(ITYPE.EQ.2 .OR. ITYPE.EQ.3)GOTO170
C READ THE LIST OF INTEGER VARIABLES.
      READ(5,1000)NINT
      READ(5,1000)(NV(K),K=1,NINT)
      WRITE(6,1010)NINT
      WRITE(6,1011)(NV(K),K=1,NINT)
      DO160K=1,NINT
      J=NV(K)
      INT(J)=K
      IF(SIGMAU(J).LE.BIGINT)GOTO160
      SIGMAU(J)=BIGINT
  160 CONTINUE
  170 DO180J=1,N
  180 ICC(J)=0
      IF(ITYPE.EQ.1 .OR. ITYPE.EQ.3)GOTO200
C READ THE LIST OF CONCAVE VARIABLES.
      READ(5,1000)NCC
      READ(5,1000)(NV(K),K=1,NCC)
      WRITE(6,1012)NCC
      WRITE(6,1013)(NV(K),K=1,NCC)
      DO190K=1,NCC
      J=NV(K)
  190 ICC(J)=K
  200 IF(ITYPE.NE.1)RETURN
C FOR THE MIXED INTEGER LINEAR PROGRAM, DETERMINE IF THE OBJECTIVE
C FUNCTION IS INTEGER VALUED.
      IROUND=0
      DO220J=1,N
      IF(INT(J).NE.0)GOTO210
      IF(C2(J).NE.0.0)RETURN
      GOTO220
  210 IC2=C2(J)
      FC2=IC2
      IF(C2(J).NE.FC2)RETURN
  220 CONTINUE
      IROUND=1
      RETURN
 1000 FORMAT(16I5)
 1001 FORMAT(6E12.0)
 1002 FORMAT(22H0TABLE OF CONSTANTS   =,6E15.6/(22X,6E15.6))
 1003 FORMAT(22H0RIGHT-HAND-SIDE      =,6E15.6/(22X,6E15.6))
 1004 FORMAT(42H0NUMBER OF VARIABLES HAVING LOWER BOUNDS =,I5)
 1005 FORMAT(22H0VARIABLES            =,16I5/(22X,16I5))
 1006 FORMAT(22H0LOWER BOUNDS         =,6E15.6/(22X,6E15.6))
 1007 FORMAT(42H0NUMBER OF VARIABLES HAVING UPPER BOUNDS =,I5)
 1008 FORMAT(22H0UPPER BOUNDS         =,6E15.6/(22X,6E15.6))
 1009 FORMAT(22H0COST COEFFICIENTS    =,6E15.6/(22X,6E15.6))
 1010 FORMAT(30H0NUMBER OF INTEGER VARIABLES =,I5)
 1011 FORMAT(22H0INTEGER VARIABLES    =,16I5/(22X,16I5))
 1012 FORMAT(30H0NUMBER OF CONCAVE VARIABLES =,I5)
 1013 FORMAT(22H0CONCAVE VARIABLES    =,16I5/(22X,16I5))
 1014 FORMAT(12H *****INPUT3)
      END
      SUBROUTINE INPUT4 (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,BORIG,RHS,
     1                   C2,C1,BI,BN,L,PJ,BINV,B,ND1,ND2,ND3,ND4,ND5,
     2                   ND6,ND7)
C ESTABLISH THE INITIAL BASIS, BASIS INVERSE, AND RIGHT-HAND-SIDE FOR
C THE LP.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IEUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION NV(ND6),IBV(ND4),NBV(ND5),IUPPER(ND5)
      DIMENSION BI(ND4),BN(ND5),U(ND6),FJ(ND4),B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1003)
C READ INITIAL FEASIBLE BASIS. INITIALIZE PARAMETERS USED IN LP.
      L10=N1-M
      DO100I=1,L10
      IUPPER(I)=0
  100 BN(I)=0.0
      IF(IFB.EQ.0)GOTO210
C INITIAL FEASIBLE BASIS PROVIDED AS INPUT.
      READ(5,1000)(IBV(I),I=1,M)
      IF(M4.EQ.0)GOTO130
      DO110I=1,M
      IF(IBV(I).GT.NM1M3)GOTO120
  110 CONTINUE
      GOTO130
C THERE ARE ARTIFICIAL VARIABLES IN THE INITIAL BASIS.
  120 IBV(MP1)=N1+1
      IBV(MP2)=N1P2
      M7=MP2
      IPHASE=2
      NPHASE=1
      CALL OBJ1 (NZ,NP,IR,IA,IS,IBV,TC,BORIG,RHS,C1,ND1,ND2,ND3,ND4,ND5)
      GOTO140
C THERE ARE NO ARTIFICIAL VARIABLES IN THE INITIAL BASIS.
  130 IBV(MP1)=N1+1
      M7=MP1
      IPHASE=1
      NPHASE=0
  140 WRITE(6,1001)(IBV(I),I=1,M7)
C FORM THE LIST OF NON-BASIC VARIABLES.
      INDEX=0
      DO160K=1,NM1M3
      DO150I=1,M
      IF(K.EQ.IBV(I))GOTO160
  150 CONTINUE
      INDEX=INDEX+1
      NBV(INDEX)=K
  160 CONTINUE
      L10=INDEX
C READ NON-BASIC VARIABLES INITIALLY AT UPPER BOUND.
      READ(5,1000)NUP
      IF(NUP.EQ.0)GOTO200
      READ(5,1000)(NV(K),K=1,NUP)
      WRITE(6,1002)(NV(K),K=1,NUP)
      DO190K=1,NUP
      INDEX=NV(K)
      DO170I=1,L10
      IF(NBV(I).EQ.INDEX)GOTO180
  170 CONTINUE
  180 IUPPER(I)=1
      BN(I)=U(INDEX)
  190 CONTINUE
  200 CONTINUE
C FORM THE BASIS INVERSE AND INITIALIZE THE RIGHT-HAND-SIDE.
      NM3M7=N+M3+M7
      CALL BINVRT (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,
     1            BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)
      M10=M7
      IEOJ=0
      NITER=0
      NBINV=0
      RETURN
C INITIAL FEASIBLE BASIS NOT PROVIDED AS INPUT.
  210 IF(M4.EQ.0)GOTO230
      M7=MP2
      LIMIT=N+M3+1
      JCOUNT=N1P2
      K=0
      DO220I=LIMIT,JCOUNT
      K=K+1
  220 IBV(K)=I
      IPHASE=2
      NPHASE=1
      CALL OBJ1 (NZ,NP,IR,IA,IS,IBV,TC,BORIG,RHS,C1,ND1,ND2,ND3,ND4,ND5)
      GOTO250
  230 M7=MP1
      LIMIT=N+M3+1
      JCOUNT=N1+1
      K=0
      DO240I=LIMIT,JCOUNT
      K=K+1
  240 IBV(K)=I
      IPHASE=1
      NPHASE=0
  250 IF(IOUTPT.GE.3)WRITE(6,1001)(IBV(I),I=1,M7)
      DO260I=1,L10
  260 NBV(I)=I
      NM3M7=N+M3+M7
      DO270I=1,M7
      DO270J=1,M7
  270 B(I,J)=0.0
      DO280K=1,M7
  280 B(K,K)=1.0
      NRH=N+M3+M7+1
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1            NRH,NZEROS)
      DO290J=1,M7
  290 BI(J)=PJ(J)
```

EXHIBIT 4 (Continued)

```
      M10=M7                                                            INPU0115
      IEOJ=0                                                            INPU0116
      NITER=0                                                           INPU0117
      NBINV=0                                                           INPU0118
      RETURN                                                            INPU0119
 1000 FORMAT(16I5)                                                      INPU0120
 1001 FORMAT(25H0INITIAL FEASIBLE BASIS =,16I5/(25X,16I5))              INPU0121
 1002 FORMAT(37H0NON-BASIC VARIABLES AT UPPER BOUND =,16I5/(37X,16I5))  INPU0122
 1003 FORMAT(12H *****INPUT4)                                           INPU0123
      END                                                               INPU0124
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE INPUT5 (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,
     1                   U,PJ,B,ND1,ND2,ND3,ND4,ND5,ND6)
C DETERMINE THE APPLICABLE LP ALGORITHM.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBLB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BIGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTOT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION IS(ND4),IBV(ND4),NBV(ND5),IUPPER(ND5)
      DIMENSION BI(ND4),U(ND6),PJ(ND4),B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1010)
C CHECK PRIMAL FEASIBILITY.
      IPRIM=0
      DO100I=1,M
      IF(BI(I).LT.EPSIM)GOTO120
  100 CONTINUE
      DO110I=1,M
      I1=IBV(I)
      IF(U(I1)-BI(I).LT.EPSIM)GOTO120
  110 CONTINUE
      IPRIM=1
  120 CONTINUE
C CHECK DUAL FEASIBILITY.
      IDUAL=0
      DO150IPOS=1,L10
      KIND=NBV(IPOS)
      DO130I=1,M7
  130 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             KIND,NZEROS)
      Q1=0.0
      DO140J1=1,NZEROS
      J=IS(J1)
  140 Q1=Q1 + B(MP1,J)*PJ(J)
      Q1=Q1 + B(MP1,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(MP1,MP2)*PJ(MP2)
      IF(IUPPER(IPOS).EQ.1)Q1=-Q1
      IF(Q1.LT.EPSIM)GOTO160
  150 CONTINUE
      IDUAL=1
  160 CONTINUE
C SELECT THE ALGORITHM TO BE USED.
      IF(IOUTPT.LE.2)GOTO170
      IF(IPHASE.EQ.2)WRITE(6,1000)
      IF(IPHASE.EQ.1)WRITE(6,1001)
  170 IF(IPRIM.EQ.0)GOTO190
      IF(IOUTPT.LE.2)GOTO180
      WRITE(6,1002)
      IF(IPHASE.EQ.2)WRITE(6,1006)
      IF(IPHASE.EQ.1)WRITE(6,1007)
  180 IALGO=1
      RETURN
```

EXHIBIT 4 (Continued)

```
 190 IF(IOUTPT.LE.2)GOTO200
     WRITE(6,1003)
 200 IF(IDUAL.EQ.0)GOTO220
     IF(IOUTPT.LE.2)GOTO210
     WRITE(6,1004)
     WRITE(6,1008)
 210 IALGO=2
     IF(IPHASE.EQ.1)RETURN
     NPHASE=2
     M10=MP1
     RETURN
 220 IF(IOUTPT.LE.2)GOTO230
     WRITE(6,1005)
     WRITE(6,1009)
 230 IALGO=0
     RETURN
1000 FORMAT(44H0THERE ARE ARTIFICIALS IN THE INITIAL BASIS.)
1001 FORMAT(47H0THERE ARE NO ARTIFICIALS IN THE INITIAL BASIS.)
1002 FORMAT(38H THE INITIAL BASIS IS PRIMAL FEASIBLE.)
1003 FORMAT(42H THE INITIAL BASIS IS NOT PRIMAL FEASIBLE.)
1004 FORMAT(36H THE INITIAL BASIS IS DUAL FEASIBLE.)
1005 FORMAT(40H THE INITIAL BASIS IS NOT DUAL FEASIBLE.)
1006 FORMAT(54H THE PRIMAL ALGORITHM (TWO PHASE METHOD) WILL BE USED.)
1007 FORMAT(54H THE PRIMAL ALGORITHM (ONE PHASE METHOD) WILL BE USED.)
1008 FORMAT(41H THE DUAL SIMPLEX ALGORITHM WILL BE USED.)
1009 FORMAT(64H NEITHER THE PRIMAL NOR THE DUAL SIMPLEX ALGORITHMS CAN
    1BE USED.)
1010 FORMAT(12H *****INPUT5)
     END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE INVERT (IS,BI,PJ,BINV,B,ND4,ND7)
C GAUSS-JORDAN METHOD OF MATRIX INVERSION.
      COMMON/P1/N,M,ITYPE,NSTRAT,NOORL1,NBVRL1,NTITE1,NOORL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLE,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBFNOD,NODE,LNODE,Z,BOUNDL,BCUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      DIMENSION IS(ND4)
      DIMENSION BI(ND4),PJ(ND4),BINV(ND7,ND7),B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1001)
C SOLVE (B)(PJ) = BI, DEVELOPING THE INVERSE OF B IN THE PROCESS.
      DO100I=1,M7
      DO100J=1,M7
  100 BINV(I,J)=0.0
      DO110K=1,M7
      BINV(K,K)=1.0
      PJ(K)=BI(K)
  110 IS(K)=0
      DO170L=1,M7
      DO120K=1,M7
      IF(IS(K).NE.0)GOTO120
      IF(ABS(B(K,L)).GT.EPSI)GOTO130
  120 CONTINUE
C CAN DROP OUT OF THIS LOOP ONLY IF A IS ILL-CONDITIONED OR SINGULAR.
      WRITE(6,1000)
      CALL EXIT
      RETURN
  130 IS(K)=L
      T=1./B(K,L)
      PJ(K)=PJ(K)*T
      DO140J=1,M7
      B(K,J)=B(K,J)*T
  140 BINV(K,J)=BINV(K,J)*T
      DO160I=1,M7
      IF(I.EQ.K)GOTO160
      T=B(I,L)
      IF(ABS(T).LE.EPSI)GOTO160
      T1=PJ(I) - T*PJ(K)
      IF(ABS(T1).LE.EPSI)T1=0.0
      PJ(I)=T1
      DO150J=1,M7
      T1=B(I,J) - T*B(K,J)
      IF(ABS(T1).LE.EPSI)T1=0.0
      B(I,J)=T1
      T1=BINV(I,J) - T*BINV(K,J)
      IF(ABS(T1).LE.EPSI)T1=0.0
  150 BINV(I,J)=T1
  160 CONTINUE
  170 CONTINUE
C BINV CONTAINS THE INVERSE OF B, UP TO A PERMUTATION OF THE ROWS.
      DO180L=1,M7
      I=IS(L)
```

EXHIBIT 4 (Continued)

```
      BI(I)=PJ(L)                                                      INVE0058
      DO180J=1,N7                                                      INVE0059
  180 B(I,J)=EINV(L,J)                                                 INVE0060
C B NOW CONTAINS THE INVERSE AND BI CONTAINS THE SOLUTICN.              INVE0061
      RETURN                                                           INVE0062
 1000 FORMAT(60H0MATRIX TO BE INVERTED IS ILL-CONDITIONED, PERHAPS SINGUINVE0063
     1LAR.)                                                            INVE0064
 1001 FORMAT(12H *****INVERT)                                          INVE0065
      END                                                              INVE0066
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE OBJ1 (NZ,NP,IR,IA,IS,IBV,TC,BORIG,RHS,C1,ND1,ND2,ND3,
     1                 ND4,ND5)
C COMPUTE AND STORE THE PHASE 1 OBJECTIVE FUNCTION.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1        NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2        ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1        NM1M3,N1P2,NP1,NSUM,NTC,M10
      DIMENSION NZ(ND1),NP(ND1),IR(ND2),IA(ND2),IS(ND4),IBV(ND4)
      DIMENSION TC(ND3),BORIG(ND4),RHS(ND4),C1(ND5)
      IF(ITRACE.GE.1)WRITE(6,1000)
      DO100K=1,M
  100 IS(K)=0
      DO110K=1,M
      IF(IBV(K).LE.NM1M3)GOTO110
      KK=IBV(K)-NM3
      IS(KK)=1
  110 CONTINUE
      DO130J=1,N
      NPOINT=NP(J)
      NZEROS=NZ(J)
      Q1=0.0
      DO120K=1,NZEROS
      NPOINT=NPOINT+1
      KK=IR(NPOINT)
      IF(IS(KK).EQ.0)GOTO120
      INDEX=IA(NPOINT)
      Q1=Q1+TC(INDEX)
  120 CONTINUE
  130 C1(J)=-Q1
      IF(M3.EQ.0)GOTO150
      KK=M1+M2
      DO140J=NP1,NM3
      KK=KK+1
      C1(J)=0.0
      IF(IS(KK).EQ.0)GOTO140
      C1(J)=1.0
  140 CONTINUE
  150 BB=0.0
      CC=0.0
      LIMIT=M1+1
      DO160I=LIMIT,M
      IF(IS(I).EQ.0)GOTO160
      BB=BB-RHS(I)
      CC=CC-BORIG(I)
  160 CONTINUE
      RHS(MP2)=BB
      BORIG(MP2)=CC
      RETURN
 1000 FORMAT(10H *****OBJ1)
      END
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE RSTART (IF,INUSE,IBV,NBV,IUPPER,IMS,F,BI,BN,B,FMS,
     1                   NI,NF,ND4,ND5,ND10,NDMS2,NDMS3,IENTRY)
C PREPARE A RESTART TAPE OR BEGIN A JOB FROM A PREVIOUSLY PREPARED
C RESTART TAPE.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBLB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      COMMON/P4/SAVE,KBRAN,X1
      DIMENSION IF(NI),INUSE(ND10),IBV(ND4),NBV(ND5),IUPPER(ND5),
     1          IMS(NDMS2)
      DIMENSION F(NF),BI(ND4),BN(ND5),B(ND4,ND4),FMS(NDMS3)
      IF(ITRACE.GE.1)WRITE(6,1002)
      IF(IENTRY.EQ.1)GOTO130
C BEGIN A JOB FROM A RESTART TAPE.
      WRITE(6,1000)
      REWIND 4
      REWIND 7
      REWIND 8
      READ(7)   M10,
     1          NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     2          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     3          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     4          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     5          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ,
     6          SAVE,KBRAN,X1
      IF(NLIST.EQ.0)GOTO110
      DO100J=1,NLIST
      READ(8)   I0,(IMS(I),I=1,NDMS2),(FMS(I),I=1,NDMS3)
      CALL WRITMS (2,IMS,NDMS2,I0)
      CALL WRITMS (3,FMS,NDMS3,I0)
  100 CONTINUE
  110 IF(2*(NODE/2).NE.NODE)GOTO120
      READ(8)   (IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),
     1          (BI(I),I=1,ND4),(BN(I),I=1,ND5),LL1,LL2,LL3,
     2          ((B(I,J),I=1,ND4),J=1,ND4)
      WRITE(4)  (IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),
     1          (BI(I),I=1,ND4),(BN(I),I=1,ND5),LL1,LL2,LL3,
     2          ((B(I,J),I=1,ND4),J=1,ND4)
  120 READ(7)   (IF(I),I=1,NI),(F(I),I=1,NF)
      REWIND 4
      REWIND 7
      REWIND 8
      RETURN
C PREPARE A RESTART TAPE.
  130 WRITE(6,1001)
      REWIND 4
      REWIND 9
      REWIND 10
      WRITE(9)  M10,
     1          NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     2              NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,      RSTA0058
     3              NBRVAR,NUPDWN,XBRNOD,IBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,  RSTA0059
     4              TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,    RSTA0060
     5              L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ,        RSTA0061
     6              SAVE,KBRAN,X1                                             RSTA0062
      WRITE(9)      (IF(I),I=1,NI),(F(I),I=1,NF)                              RSTA0063
      IF(NLIST.EQ.0)GOTO150                                                   RSTA0064
      DO140I0=1,MXLIST                                                        RSTA0065
      IF(INUSE(I0).EQ.0)GOTO140                                               RSTA0066
      CALL READMS (2,IMS,NDMS2,I0)                                            RSTA0067
      CALL READMS (3,FMS,NDMS3,I0)                                            RSTA0068
      WRITE(10) I0,(IMS(I),I=1,NDMS2),(FMS(I),I=1,NDMS3)                      RSTA0069
  140 CONTINUE                                                                RSTA0070
  150 IF(2*(NODE/2).NE.NODE)GOTO160                                           RSTA0071
      READ(4)       (IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),    RSTA0072
     1              (BI(I),I=1,ND4),(BN(I),I=1,ND5),LL1,LL2,LL3,              RSTA0073
     2              ((B(I,J),I=1,ND4),J=1,NC4)                                RSTA0074
      WRITE(10)     (IBV(I),I=1,ND4),(NBV(I),I=1,ND5),(IUPPER(I),I=1,ND5),    RSTA0075
     1              (BI(I),I=1,ND4),(BN(I),I=1,ND5),LL1,LL2,LL3,              RSTA0076
     2              ((B(I,J),I=1,ND4),J=1,NC4)                                RSTA0077
  160 END FILE 9                                                              RSTA0078
      END FILE 10                                                             RSTA0079
      CALL EXIT                                                               RSTA0080
      RETURN                                                                  RSTA0081
 1000 FORMAT(38H0BEGINNING THE JOB FROM RESTART TAPES.)                       RSTA0082
 1001 FORMAT(24H0RESTART TAPES PREPARED.)                                     RSTA0083
 1002 FORMAT(12H *****RSTART)                                                 RSTA0084
      END                                                                     RSTA0085
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE SIMPLE (NZ,NP,IR,IA,IS,NV,IBV,NBV,IUPPER,TC,RHS,C2,C1,
     1                  BI,BN,U,PJ,BINV,XJ,V,XZ,B,ND1,ND2,ND3,ND4,ND5,
     2                  ND6,ND7)
C PRIMAL AND DUAL SIMPLEX ALGORITHMS FOR SOLVING THE LP.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,UNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBRNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IECJ
      DIMENSION IS(ND4),NV(ND6),IBV(ND4),NBV(ND5),IUPPER(ND5)
      DIMENSION BI(ND4),BN(ND5),U(ND6),PJ(ND4),XJ(ND4),V(ND6),XZ(ND6),
     1          B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1022)
      IF(IOUTPT.LE.1)GOTO110
      IF(IALGO.EQ.2)GOTO100
      WRITE(6,1000)
      IF(NODE.NE.1)GOTO110
      IF(IPHASE.EQ.1)WRITE(6,1001)
      IF(IPHASE.EQ.2)WRITE(6,1002)NPHASE
      GOTO110
  100 WRITE(6,1003)
  110 ITER=0
C***********************************************************************
C INCREMENT THE ITERATION COUNTER. REINVERT THE BASIS MATRIX IF
C NECESSARY.
C***********************************************************************
  120 ITER=ITER+1
      NITER=NITER+1
      NBINV=NBINV+1
      IF(ITER.GT.MXITER)GOTO630
      IF(NBINV.NE.MBINV)GOTO160
C REINVERT THE BASIS MATRIX EVERY MBINV ITERATIONS.
      NBINV=0
      IF(IOUTPT.GE.2)WRITE(6,1004)NITER
      IF(IOUTPT.LE.4)GOTO140
      WRITE(6,1005)
      DO130I=1,M7
  130 WRITE(6,1006)IBV(I),(B(I,J),J=1,M7),BI(I)
  140 CALL BINVRT (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,BI,U,PJ,
     1             BINV,B,ND1,ND2,ND3,ND4,ND5,ND6,ND7)
      IF(IOUTPT.LE.4)GOTO160
      WRITE(6,1005)
      DO150I=1,M7
  150 WRITE(6,1006)IBV(I),(B(I,J),J=1,M7),BI(I)
  160 GOTO(170,300),IALGO
C***********************************************************************
C PRIMAL ALGORITHM.
C PRICE-OUT THE NON-BASIC VARIABLES. THE ENTERING VARIABLE IS THE FIRST
C ONE ENCOUNTERED HAVING NEGATIVE REDUCED COST.
C***********************************************************************
  170 DO200IPOS=1,L10
      KIND=NBV(IPOS)
      DO180I=1,M7
  180 PJ(I)=0.0
      CALL GETCOL (MZ,MP,IR,IA,IS,TC,RHS,C2,C1,PJ,MD1,MD2,MD3,MD4,MD5,
     1              KIND,NZEROS)
      Q1=0.
      DO190J1=1,NZEROS
      J=IS(J1)
  190 Q1=Q1 + B(M10,J)*PJ(J)
      Q1=Q1 + B(M10,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(M10,MP2)*PJ(MP2)
C Q1 IS THE REDUCED COST FOR VARIABLE J=KIND.
      IF(IUPPER(IPOS).EQ.1)Q1=-Q1
      IF(Q1.LT.EPSIM*1000.)GOTO240
  200 CONTINUE
C FOR ALL NON-BASIC VARIABLES, THE REDUCED COST IS NON-NEGATIVE.
      IF(IPHASE.NE.2)GOTO640
      IF(NPHASE.EQ.2)GOTO640
C CHECK IF ANY ARTIFICIALS REMAIN IN THE BASIS.
      INDEX=0
      DO210I=1,M7
      J=IBV(I)
      IF(J.GT.NM1M3 .AND. J.LE.N1)INDEX=1
  210 CONTINUE
      IF(BI(MP2).GE.EPSIM*1000.)GOTO220
      IF(INDEX.EQ.1)GOTO580
C PHASE 1 TERMINATES. ENTER PHASE 2.
  220 IF(IOUTPT.GE.2)WRITE(6,1007)NITER
      M10=M10-1
      IF(INDEX.EQ.1)GOTO230
C THERE ARE NO ARTIFICIALS IN THE BASIS. DELETE THE PHASE 1 OBJECTIVE
C FUNCTION FROM THE PROGRAM.
      M7=MP1
      IPHASE=1
      NPHASE=0
      NM3M7=N+M3+M7
      GOTO120
C THERE ARE ARTIFICIALS IN THE BASIS. MAINTAIN THE PHASE 1 OBJECTIVE
C FUNCTION AS A CONSTRAINT.
  230 NPHASE=2
      GOTO120
C***********************************************************************
C PRIMAL ALGORITHM.
C SELECT THE LEAVING BASIC VARIABLE.
C***********************************************************************
C COMPUTE THE UPDATED COLUMN, XJ = B-INVERSE * PJ.
  240 DO260I=1,M7
      Q1=0.0
      DO250J1=1,NZEROS
      J=IS(J1)
  250 Q1=Q1 + B(I,J)*PJ(J)
      Q1=Q1 + B(I,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(I,MP2)*PJ(MP2)
  260 XJ(I)=Q1
C COMPUTE THE MINIMUM OVER XJ(I).GT.0 OF THE BASIC VARIABLE VALUES
C DIVIDED BY XJ(I).
      XPIV=BIGN
      DO280I=1,M7
      IF(I.EQ.MP1)GOTO280
      Q1=XJ(I)
      IF(IUPPER(IPOS).EQ.1)Q1=-Q1
C Q1 IS XJ(I).
      IF(Q1.LT.EPSI)GOTO270
      Q2=BI(I)
C Q2 IS THE BASIC VARIABLE VALUE.
      IF(Q2/Q1.GE.XPIV)GOTO280
      KPIV=I
C KPIV INDICATES THE LEAVING VARIABLE.
      XPIV=Q2/Q1
      ITHIA=1
C ENTERING VARIABLE DOES NOT FORCE ANY VARIABLE TO ITS UPPER BOUND
C (ITHIA = 1).
      GOTO280
  270 Q1=-Q1
      IF(Q1.LT.EPSI)GOTO280
      I1=IBV(I)
      Q2=U(I1)-BI(I)
      IF(Q2/Q1.GE.XPIV)GOTO280
      KPIV=I
      XPIV=Q2/Q1
      ITHIA=2
C ENTERING VARIABLE FORCES LEAVING VARIABLE TO ITS UPPER BOUND
C (ITHIA = 2).
  280 CONTINUE
      IF(U(KIND).GE.XPIV)GOTO290
      ITHIA=3
C ENTERING VARIABLE ENTERS AT ITS UPPER BOUND (ITHIA = 3).
      XPIV=U(KIND)
  290 IF(XPIV.GE.BIGN)GOTO610
      IBVK=IBV(KPIV)
      GOTO440
C*******************************************************************
C DUAL SIMPLEX ALGORITHM.
C SELECT THE LEAVING BASIC VARIABLE.
C*******************************************************************
  300 IF(IBUBOP.EQ.1)GOTO310
C TEST THE OBJECTIVE VALUE AGAINST THE BEST UPPER BOUND.
      IF(-BI(MP1)+TSIG.GE.(1.-TOL1)*UNOT)GOTO620
C COMPUTE THE MINIMUM OF THE RIGHT-HAND-SIDES.
  310 BMIN=BIGN
      KPIV=0
      DO320I=1,M7
      IF(I.EQ.MP1)GOTO320
      IF(BI(I).GE.BMIN)GOTO320
      BMIN=BI(I)
      KPIV=I
  320 CONTINUE
      JPIV=0
      DO330I=1,M7
      IF(I.EQ.MP1)GOTO330
      I1=IBV(I)
      IF(U(I1)-BI(I).GE.BMIN)GOTO330
      BMIN=U(I1)-BI(I)
      JPIV=I
  330 CONTINUE
```

```
C IF THIS MINIMUM IS NONNEGATIVE, WE ARE AT THE OPTIMUM.
      IF(BMIN.GE.EPSIM*1000.)GOTO640
      ITHIA=1
      IF(JPIV.EQ.0)GOTO340
      ITHIA=2
      KPIV=JPIV
  340 IBVK=IBV(KPIV)
C***********************************************************************
C DUAL SIMPLEX ALGORITHM.
C SELECT THE ENTERING VARIABLE.
C***********************************************************************
C COMPUTE THE MINIMUM OVER XJ(I).LT.0 OF THE DUAL VARIABLE VALUES
C DIVIDED BY -XJ(I).
      XPIV=BIGN
      KIND=0
      DO400JPOS=1,L10
      JIND=NBV(JPOS)
      DO350I=1,M7
  350 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             JIND,NZEROS)
      Q1=0.0
      DO360J1=1,NZEROS
      J=IS(J1)
  360 Q1=Q1 + B(KPIV,J)*PJ(J)
      Q1=Q1 + B(KPIV,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(KPIV,MP2)*PJ(MP2)
      IF(ITHIA.EQ.2)GOTO370
      IF(IUPPER(JPOS).EQ.0)Q1=-Q1
      GOTO380
  370 IF(IUPPER(JPOS).EQ.1)Q1=-Q1
  380 IF(Q1.LE.EPSI)GOTO400
C Q1 IS -XJ(I).
      Q2=0.0
      DO390J1=1,NZEROS
      J=IS(J1)
  390 Q2=Q2 + B(MP1,J)*PJ(J)
      Q2=Q2 + B(MP1,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q2=Q2 + B(MP1,MP2)*PJ(MP2)
      IF(IUPPER(JPOS).EQ.1)Q2=-C2
C Q2 IS THE DUAL VARIABLE VALUE.
      IF(Q2/Q1.GE.XPIV)GOTO400
      XPIV=Q2/Q1
      KIND=JIND
      IPOS=JPOS
  400 CONTINUE
C IF THERE IS NO PIVOT ELEMENT, THE PRIMAL PROGRAM IS INFEASIBLE
C (THE DUAL PROGRAM IS UNBOUNDED).
      IF(KIND.EQ.0)GOTO580
C COMPUTE THE UPDATED COLUMN, XJ = B-INVERSE * PJ.
      DO410I=1,M7
  410 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             KIND,NZEROS)
      DO430I=1,M7
      Q1=0.0
      DO420J1=1,NZEROS
```

EXHIBIT 4 (Continued)

```
      J=IS(J1)                                                          SIMP0229
  420 Q1=Q1 + B(I,J)*PJ(J)                                              SIMP0230
      Q1=Q1 + B(I,MP1)*PJ(MP1)                                          SIMP0231
      IF(IPHASE.EQ.2)Q1=Q1 + B(I,MP2)*PJ(MP2)                           SIMP0232
  430 XJ(I)=Q1                                                          SIMP0233
C***********************************************************************SIMP0234
C PRIMAL AND DUAL SIMPLEX ALGORITHMS.                                   SIMP0235
C PIVOT THE ENTERING AND LEAVING VARIABLES.                             SIMP0236
C***********************************************************************SIMP0237
  440 IF(IOUTPT.EQ.5)WRITE(6,1008)NITER,KIND,IBVK,ITHIA                 SIMP0238
      INDEX=0                                                           SIMP0239
      IF(IUPPER(IPOS).EQ.1)GOTO480                                      SIMP0240
      GOTO(510,470,450),ITHIA                                           SIMP0241
  450 IUPPER(IPCS)=1                                                    SIMP0242
      BN(IPOS)=L(KIND)                                                  SIMP0243
      DO460I=1,M7                                                       SIMP0244
  460 BI(I)=BI(I) - U(KIND)*XJ(I)                                       SIMP0245
      GOTO120                                                           SIMP0246
  470 BI(KPIV)=BI(KPIV) - U(IBVK)                                       SIMP0247
      IUPPER(IPCS)=1                                                    SIMP0248
      BN(IPOS)=U(IBVK)                                                  SIMP0249
      GOTO510                                                           SIMP0250
  480 IUPPER(IPCS)=0                                                    SIMP0251
      BN(IPOS)=0.0                                                      SIMP0252
C CHANGE THE RIGHT-HAND-SIDE AFTER THE PIVOT.                           SIMP0253
      INDEX=1                                                           SIMP0254
      GOTO(510,470,490),ITHIA                                           SIMP0255
  490 DO500I=1,M7                                                       SIMP0256
  500 BI(I)=BI(I) + U(KIND)*XJ(I)                                       SIMP0257
      GOTO120                                                           SIMP0258
  510 T=1./XJ(KPIV)                                                     SIMP0259
      BI(KPIV)=BI(KPIV)*T                                               SIMP0260
      DO520K=1,M7                                                       SIMP0261
  520 B(KPIV,K)=B(KPIV,K)*T                                             SIMP0262
      DO540J=1,M7                                                       SIMP0263
      IF(J.EQ.KPIV)GOTO540                                              SIMP0264
      T=XJ(J)                                                           SIMP0265
      IF(ABS(T).LE.EPSI)GOTO540                                         SIMP0266
      T1=BI(J) - T*BI(KPIV)                                             SIMP0267
      IF(ABS(T1).LE.EPSI)T1=0.0                                         SIMP0268
      BI(J)=T1                                                          SIMP0269
      DO530K=1,M7                                                       SIMP0270
      T1=B(J,K) - T*B(KPIV,K)                                           SIMP0271
      IF(ABS(T1).LE.EPSI)T1=0.0                                         SIMP0272
  530 B(J,K)=T1                                                         SIMP0273
  540 CONTINUE                                                          SIMP0274
      IF(INDEX.EQ.0)GOTO550                                             SIMP0275
      BI(KPIV)=BI(KPIV) + U(KIND)                                       SIMP0276
C UPDATE THE NBV AND IBV ARRAYS.                                        SIMP0277
  550 I=IBV(KPIV)                                                       SIMP0278
      IBV(KPIV)=NBV(IPOS)                                               SIMP0279
      IF(I.GT.NM1M3 .AND. I.LE.N1)GOTO560                               SIMP0280
      IF(IALGO.EQ.2 .AND. I.EQ.N1P2)GOTO0560                            SIMP0281
      NBV(IPOS)=I                                                       SIMP0282
      GOTO120                                                           SIMP0283
  560 NBV(IPOS)=NBV(L10)                                                SIMP0284
      IUPPER(IPCS)=IUPPER(L10)                                          SIMP0285
```

EXHIBIT 4 (Continued)

```
      BN(IPOS)=BN(L10)
      L10=L10-1
      IF(NPHASE.EQ.1)GOTO120
      DO570I=1,M7
      J=IBV(I)
      IF(J.GT.NM1M3 .AND. J.LE.N1)GOTO120
  570 CONTINUE
C THERE ARE NO ARTIFICIALS IN THE BASIS. DELETE THE PHASE 1 OBJECTIVE
C FUNCTION FROM THE PROGRAM.
      M7=MP1
      IPHASE=1
      NPHASE=0
      NM3M7=N+M3+M7
      GOTO120
C***********************************************************************
C FINAL OUTPUT.
C***********************************************************************
  580 IF(IOUTPT.EQ.0)GOTO600
      WRITE(6,1009)
      IF(IOUTPT.LE.2)GOTO600
      IF(IALGO.EQ.1)GOTO590
      WRITE(6,1010)IBVK,BMIN
  590 WRITE(6,1011)(IBV(I),I=1,M7)
      WRITE(6,1012)(BI(I),I=1,M7)
  600 IEOJ=1
      GOTO640
  610 WRITE(6,1013)
      WRITE(6,1014)KINO
      WRITE(6,1011)(IBV(I),I=1,M7)
      WRITE(6,1015)(XJ(I),I=1,M7)
      WRITE(6,1012)(BI(I),I=1,M7)
      IEOJ=2
      GOTO640
  620 IF(IOUTPT.NE.0)WRITE(6,1016)
      IEOJ=3
      GOTO640
  630 WRITE(6,1017)
      IEOJ=4
  640 DO650I=1,N1P2
  650 XZ(I)=0.0
      DO660I=1,NM3M7
      NV(I)=0
  660 V(I)=0.0
      DO680K=1,M7
      I=IBV(K)
      NV(I)=I
      IF(K.EQ.MP1)GOTO670
      V(I)=BI(K)
      XZ(I)=BI(K)
      GOTO680
  670 V(I)=-BI(K)
      XZ(I)=-BI(K)
  680 CONTINUE
      Z=-BI(MP1)
      IF(IOUTPT.LE.3)GOTO710
      DO700K=1,M7
      DO690I=K,NM3M7
```

EXHIBIT 4 (Continued)

```
      IF(NV(I).EQ.0)GOTO690
      IF(I.EQ.K)GOTO700
      NV(K)=NV(I)
      V(K)=V(I)
      NV(I)=0
      GOTO700
  690 CONTINUE
  700 CONTINUE
      WRITE(6,1018)
      WRITE(6,1019)(NV(K),V(K),K=1,M7)
  710 CONTINUE
      DO720I=1,NM3M7
      NV(I)=0
  720 V(I)=0.0
      KK=0
      DO730K=1,L10
      IF(IUPPER(K).EQ.0)GOTO730
      KK=KK+1
      I=NBV(K)
      NV(I)=I
      V(I)=BN(K)
      XZ(I)=BN(K)
  730 CONTINUE
      IF(IOUTPT.LE.3)GOTO780
      IF(KK.EQ.0)GOTO760
      DO750K=1,KK
      DO740I=K,NM3M7
      IF(NV(I).EQ.0)GOTO740
      IF(I.EQ.K)GOTO750
      NV(K)=NV(I)
      V(K)=V(I)
      NV(I)=0
      GOTO750
  740 CONTINUE
  750 CONTINUE
      WRITE(6,1020)
      WRITE(6,1019)(NV(K),V(K),K=1,KK)
  760 IF(IOUTPT.LE.4)GOTO780
      WRITE(6,1005)
      DO770I=1,M7
  770 WRITE(6,1006)IBV(I),(B(I,J),J=1,M7),BI(I)
  780 IF(IOUTPT.GE.2)WRITE(6,1021)ITER,NITER
      ITRTOT=ITRTOT + ITER
      IF(NITER.GT.ITRMAX)ITRMAX=NITER
      RETURN
 1000 FORMAT(17H0PRIMAL ALGORITHM)
 1001 FORMAT(17H ONE PHASE METHOD)
 1002 FORMAT(47H TWO PHASE METHOD - BEGIN COMPUTATIONS IN PHASE,I2)
 1003 FORMAT(23H0DUAL SIMPLEX ALGORITHM)
 1004 FORMAT(32H0BASIS REINVERTED ON ITERATION =,I5)
 1005 FORMAT(48H0BASIC VARIABLES/BASIS INVERSE/RIGHT-HAND-SIDE =)
 1006 FORMAT(1H0,I5,8E15.6/(6X,8E15.6))
 1007 FORMAT(27H ENTER PHASE 2 ON ITERATION,I5)
 1008 FORMAT(1X,I5,21H. ENTERING VARIABLE =,I5,20H, LEAVING VARIABLE =,
     1       I5,16H, THETA = THETA(,I1,1H))
 1009 FORMAT(34H0THE PRIMAL PROGRAM IS INFEASIBLE.)
 1010 FORMAT(20H0LEAVING VARIABLE  =,I5/
```

EXHIBIT 4 (Continued)

```
     1        20HORIGHT-HAND-SIDE    =,E15.6)                            SIMP0400
1011 FORMAT(20H0BASIC VARIABLES      =,16I5/(20X,16I5))                  SIMP0401
1012 FORMAT(20H0RIGHT-HAND-SIDE    =,6E15.6/(20X,6E15.6))                SIMP0402
1013 FORMAT(43H0THE PRIMAL PROGRAM HAS UNBOUNDED SOLUTION.)              SIMP0403
1014 FORMAT(20H0ENTERING VARIABLE =,I5)                                  SIMP0404
1015 FORMAT(20H0UPDATED COLUMN      =,6E15.6/(20X,6E15.6))               SIMP0405
1016 FORMAT(45H0THE DUAL VALUE EXCEEDS THE BEST UPPER BOUND.)            SIMP0406
1017 FORMAT(55H0THE MAXIMUM NUMBER OF LP ITERATIONS HAS BEEN EXCEEDED.)SIMP0407
1018 FORMAT(16H0BASIC VARIABLES)                                         SIMP0408
1019 FORMAT(17H0 VARIABLE/VALUE =,5(I5,E15.6)/(17X,5(I5,E15.6)))         SIMP0409
1020 FORMAT(35H0NON-BASIC VARIABLES AT UPPER BOUND)                      SIMP0410
1021 FORMAT(43H0NUMBER OF LP ITERATIONS THIS COMPUTATION =,I5,           SIMP0411
     1        14H, CUMULATIVE =,I5)                                      SIMP0412
1022 FORMAT(12H *****SIMPLE)                                             SIMP0413
     END                                                                 SIMP0414
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE SLOPES (NZ,NP,IR,IA,IS,IBV,NBV,IUPPER,TC,RHS,C2,C1,PJ,
     1                   XJ,S0,S1,B,ND1,ND2,ND3,ND4,ND5,ND8)
C DETERMINE THE LEFT AND RIGHT SLOPES ASSOCIATED WITH THE OPTIMAL
C SOLUTION VALUE AS A FUNCTION OF A PARAMETER.
      COMMON/P1/N,M,ITYPE,NSTRAT,NODRL1,NBVRL1,NTITE1,NODRL2,NBVRL2,
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)
      COMMON/P2/EPSI,EPSIM,BIGN,BEGIM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2,
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10
      COMMON/P3/NODNOT,LNOT,IBUBOP,LPHASE,NODRUL,NBVRUL,NTIGHT,NLIST,
     1          NLISTS,NFEAS,LSTMX,ITRTCT,ITRMAX,BLB,NBRNOD,PBRNOD,
     2          NBRVAR,NUPDWN,XBRNOD,TBFNOD,NODE,LNODE,Z,BOUNDL,BOUNDU,
     3          TSIG,IFEAS,IBRVR1,IUPDN1,XBRVR1,IBRVR2,IUPDN2,XBRVR2,
     4          L10,NITER,NBINV,M7,IPHASE,NPHASE,NM3M7,IALGO,IEOJ
      DIMENSION IS(ND4),IBV(ND4),NBV(ND5),IUPPER(ND5)
      DIMENSION PJ(ND4),XJ(ND4),S0(ND8),S1(ND8),B(ND4,ND4)
      IF(ITRACE.GE.1)WRITE(6,1002)
      DO100I=1,M7
      IF(I.EQ.MP1)GOTO100
      S0(I)=-BIGN
      S1(I)=BIGN
  100 CONTINUE
      DO150IPOS=1,L10
      KIND=NBV(IPOS)
      DO110I=1,M7
  110 PJ(I)=0.0
      CALL GETCOL (NZ,NP,IR,IA,IS,TC,RHS,C2,C1,PJ,ND1,ND2,ND3,ND4,ND5,
     1             KIND,NZEROS)
      DO130I=1,M7
      Q1=0.0
      DO120J1=1,NZEROS
      J=IS(J1)
  120 Q1=Q1 + B(I,J)*PJ(J)
      Q1=Q1 + B(I,MP1)*PJ(MP1)
      IF(IPHASE.EQ.2)Q1=Q1 + B(I,MP2)*PJ(MP2)
      IF(IUPPER(IPOS).EQ.1)Q1=-Q1
  130 XJ(I)=Q1
      IF(XJ(MP1).LE.EPSI)XJ(MP1)=0.0
      DO150I=1,M7
      IF(I.EQ.MP1)GOTO150
      IF(ABS(XJ(I)).LE.EPSI)GOTO150
      X0=XJ(MP1)/(-XJ(I))
      IF(XJ(I).LT.EPSIM)GOTO140
C S0(I) IS THE MAXIMUM OVER X(I,J).GT.0 OF THE REDUCED COST DIVIDED
C BY -X(I,J).
      IF(X0.LE.S0(I))GOTO150
      S0(I)=X0
      GOTO150
C S1(I) IS THE MINIMUM OVER X(I,J).LT.0 OF THE REDUCED COST DIVIDED
C BY -X(I,J).
  140 IF(X0.GE.S1(I))GOTO150
      S1(I)=X0
  150 CONTINUE
      IF(IOUTPT.LE.2)RETURN
      WRITE(6,1000)
      DO160I=1,M7
      IF(I.EQ.MP1)GOTO160
```

EXHIBIT 4 (Continued)

```
      WRITE(6,1001)IBV(I),S0(I),S1(I)                                   SLOP0058
  160 CONTINUE                                                          SLOP0059
      RETURN                                                            SLOP0060
 1000 FORMAT(1H0,2X,5HBASIC,11X,4HLEFT,12X,5HRIGHT/                     SLOP0061
     1       1X,8HVARIABLE,9X,5HSLOPE,12X,5HSLOPE//)                    SLOP0062
 1001 FORMAT(3X,I5,3X,E15.6,2X,E15.6)                                   SLOP0063
 1002 FORMAT(12H *****SLOPES)                                           SLOP0064
      END                                                               SLOP0065
```

EXHIBIT 4 (Continued)

```
      SUBROUTINE TIMEC                                                  TIME0001
C PRINT THE ELAPSED TIME SINCE THE BEGINNING OF THIS JOB.                TIME0002
      COMMON/P1/N,M,ITYPE,NSTRAT,NOORL1,NBVRL1,NTITE1,NOORL2,NBVRL2,    TIME0003
     1          NTITE2,MXLIST,LISTOP,ITAPE,IFB,MXITER,MBINV,IOUTPT,     TIME0004
     2          ITRACE,MSTART,TIME1,TOL1,TOL2,PCBUB,ALPHA(10)           TIME0005
      COMMON/P2/EPSI,EPSIM,BIGN,BEGTM,M1,M2,M3,M4,N1,MP1,MP2,NM3,NM1M2, TIME0006
     1          NM1M3,N1P2,NP1,NSUM,NTC,M10                             TIME0007
      IF(ITRACE.GE.1)WRITE(6,1001)                                      TIME0008
      CALL SECOND (X)                                                   TIME0009
      X=X-BEGTM                                                         TIME0010
      WRITE(6,1000)X                                                    TIME0011
      RETURN                                                            TIME0012
 1000 FORMAT(7H0TIME =,F9.3,8H SECONDS)                                 TIME0013
 1001 FORMAT(11H *****TIMEC)                                            TIME0014
      END                                                               TIME0015
```

APPENDIX G

REFERENCES

1. Robert S. Garfinkel and George L. Nemhauser, Integer Programming, John Wiley and Sons, Inc., New York, 1972.

2. Frederick S. Hillier and Gerald J. Lieberman, Introduction to Operations Research, Holden-Day, Inc., San Francisco, 1967.

3. Leon S. Lasdon, Optimization Theory for Large Systems, The MacMillan Company, London, 1970.

APPENDIX H

DISTRIBUTION

Defense Documentation Center Cameron Station Alexandria, Virginia 22314	(12)
Library of Congress Attn: Gift and Exchange Division Washington, D. C. 20540	(4)

Local:

K30	(25)
K70	(1)
E41	(2)
X210	(1)
X2101(GIDEP)	(5)